AD-A106 176 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/G 20/8
FEASIBILITY OF INTERFACING A MICROCOMPUTER WITH A MULTICHANNEL --ETC(U)
MAR 81 D CLARKE
UNCLASSIFIED AFIT/GNE/PH/81M-2 NL

1 OF 2
AD
A106176

AFIT/GNE/PH/81M-2

LEVEL II

FEASIBILITY OF INTERFACING
A MICROCOMPUTER
WITH A MULTICHANNEL ANALYZER
TO PERFORM GAMMA RAY SPECTROSCOPY.

THESIS

AFIT/GNE/PH/81M-2 DeFrance Clarke, III
Captain USAF

Approved for public release; distribution unlimited

AFIT/GNE/PH/81M-2

FEASIBILITY OF INTERFACING A MICROCOMPUTER WITH A MULTICHANNEL ANALYZER TO PERFORM GAMMA RAY SPECTROSCOPY

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air Training Command
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

by

DeFrance Clarke III, B.S., M.S.
Captain USAF
Graduate Nuclear Engineering

March 1981

Accession For	
NTIS GRA&I	X
DTIC TAB	
Unannounced	
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A	

Approved for public release; distribution unlimited

Preface

This project was instigated by my thesis advisor, Dr. George John, who saw the need to upgrade the equipment in the Nuclear Laboratory at the Air Force Institute of Technology. I have been fascinated with computers since my undergraduate college years and eagerly accepted the challenge to develop hardware and software for this system. My limited knowledge of digital hardware was augmented by the advice of Capt. David Hardin. Many members and students of the Electrical Engineering department helped as I was assembling and testing hardware, but I wish to single out Dr. Thomas Hartrum, who provided access to the equipment used, and Mr. Daniel Zambon who provided detailed advice, suggestions, and a hot soldering iron on many occasions.

The primary factor which limited the scope of this study was the availability of only 4096 bytes of program storage. Most of the software written and time spent debugging was devoted to making even a limited system fit in the program space available.

DeFrance Clarke III

Contents

	Page
Preface	ii
List of Figures	v
List of Tables	vi
Abstract	vii
I. Introduction	1
Background	1
Statement of the Problem	3
Requirements	3
Scope	6
Assumptions	6
Approach	6
Sequence of Presentation	7
II. Demonstration System	8
Hardware	8
Multichannel Analyzer	8
Microcomputer	14
Interface Board	16
Random Access Memory Board	18
Terminal	22
Hardware Summary	22
Software	22
Functions	23
Source Languages	23
Modules	24
Evaluation of the Demonstration System	36
Performance	36
Timing	37
Memory	37
Summary	38

Contents (Continued)

	Page
III. Proposed System	40
Hardware	40
Microcomputer	40
Memory	41
Disk Drives	41
High Speed Math Unit	43
Terminal	43
Software	45
General Considerations	45
Transfer Functions	47
Manipulation Functions	48
Peak Processing Functions	49
Calibration Functions	50
Display Function	50
Report Function	51
Acquire Function	51
Summary	52
Cost Benefit Comparison	52
Cost of Proposed System	52
Commercial Multichannel Analyzers	53
Comparison	54
IV. Conclusions and Recommendations	56
Discussion	56
Conclusions	57
Recommendations	58
Bibliography	59
Appendix A: Program Listings	61
ASSEM Module	62
ENCAL Module	72
GAMMA Module	74
GETDAT Module	75
INT4 Module	81
MONIT Module	93
PKPROC Fortran Module	102
PKPROC Assembly Module	104
PKSRCH Module	116
Appendix B: Interface Testing	118
Vita	129

List of Figures

Number	Title	Page
1	Multichannel Analyzer Mode Switch	9
2	Interface Board Block Diagram	17
3	Interface Card Layout	19
4	Interface Board Input Schematic	20
5	Interface Board Schematic for Output, Interupt, and Devices	21
6	Processing a Typical Peak	28

List of Tables

Number	Title	Page
I	Multichannel Analyzer Signal Lines PRINTER A Connector	11
II	Multichannel Analyzer Signal Lines PRINTER B Connector	12
III	Multichannel Analyzer Signal Lines EXTERNAL Connector	13
IV	Demonstration System Program Structure	25
V	32-Bit Integer Arithmetic Routines of Module INT4	32
VI	Catalog Prices of Proposed System Hardware	54

Abstract

Interfacing a microcomputer to a Nuclear Data Series 2200 multichannel analyzer has been demonstrated. Programs for data transfer, peak detection, peak net area integration, peak centroid location, and energy calibration were written. Data transfer and peak detection scan were demonstrated on 4096 channel spectra in less than 5 minutes. Multibyte integer arithmetic routines for the 8080 microprocessor were written to conserve space and time. A proposed system for gamma ray spectrum analysis is described in hardware and software terms. The hardware for the proposed system would cost less than $9000 while commercial multichannel analyzers with the same capabilities would cost in excess of $23,000.

FEASIBILITY OF INTERFACING A MICROCOMPUTER
WITH A MULTICHANNEL ANALYZER
TO PERFORM GAMMA RAY SPECTROSCOPY

I. Introduction

This report presents the results of a study which investigated the feasibility of using a microcomputer attached to an existing multichannel analyzer to analyze gamma ray spectra. Hardware and software options were explored.

Background

Gamma ray spectroscopy is the process of analyzing the energy spectrum of a gamma ray source. The source emits gamma rays of varying energy; some of these rays are absorbed in a detector and produce electrical pulses with amplitude proportional to the energy of the gamma ray absorbed. The electrical pulses are amplified and fed into a multichannel analyzer which measures the pulse amplitude and increments one of its channels. The number of the channel incremented is proportional to the size of the pulse and the gain setting of the multichannel analyzer. After accumulating pulses for a preset period of time, the values of the channels are read out in sequential order. The results are usually displayed on an oscilloscope with channel number on the x-axis and counts per channel on the y-axis. When displayed in this manner, the oscilloscope trace characteristically has peaks. Some of these peaks correspond to the energy of a particular nuclear transition. The net number of counts in the channels defining a peak is proportional to the intensity of that gamma ray and the efficiency of the detection system at that energy. The number of channels between two peaks is proportional to the difference in energy between the gamma rays.

The desired results of gamma ray spectroscopy include the determination of gamma ray energy and intensity for each of the peaks displayed. These results are usually obtained by analyzing a spectrum of a source whose activity and decay scheme are known, using the results to calibrate the system, and measuring the activity of an unknown sample with the calibrated system. For each of these steps, it is necessary to compute the location and area of each peak. Automation of this tedious task is necessary for all but trivial studies.

Currently, in the Nuclear Laboratory at the Air Force Institute of Technology, spectra are analyzed by writing the channel contents on magnetic tape, taking the tape to the computer center, and using large mainframe computers (such as the Cyber 74) to perform the calculations. The disadvantage of this procedure is that it takes one to three days before results are available. Increasing contention for the limited computer resources (tape drive, disk memory, central processor time, terminal, and plotter) will increase the time required to obtain results. Faster response by a dedicated system can improve the experimental conditions by allowing changes to be made in the experimental setup based on early results. A dedicated system would also be of more pedagogic value because a student could receive nearly immediate feedback (5-10 minutes) and be able to recognize and correct mistakes shortly after they were made. Furthermore, after graduation, a student is likely to find gamma ray systems which perform most or all of the spectrum analysis rather than the current system which only gathers data. A dedicated system could be provided either by purchasing a new multichannel analyzer with the required capabilities or by purchasing a microcomputer, interfacing it to the existing multichannel analyzer, and writing programs which will analyze gamma-ray spectra.

Statement of the Problem

The purpose of this effort is to determine whether it is feasible, economical, and worthwhile to interface a microcomputer to the existing multichannel analyzer and use the microcomputer to analyze gamma ray spectra. The primary measure of feasibility is the time required to analyze a typical spectrum. A secondary purpose is to determine the necessary and desirable characteristics of such a microcomputer system.

Requirements

Before a system can be designed or evaluated, the required and desirable performance of that system must be defined. The following two paragraphs provide the rationale and specify the required functions, maximum time allowed, and the necessity of program changes for a system which performs analysis of gamma ray spectra at the Air Force Institute of Technology.

The multichannel analyzer is primarily used to teach graduate nuclear engineering students the basic techniques of gamma ray spectroscopy in a laboratory class which meets twice a week for three hours. An improved system must provide hands on training to several students in the three hour period. At a minimum an analysis of a spectrum must provide for

- scanning a 4096 channel spectrum and identifying peaks,
- locating each peak centroid within less than one channel,
- integrating the net area of each peak after background counts have been subtracted,
- calibrating the analyzer in energy and efficiency, and
- applying the calibration to an unknown spectrum to determine the energy and intensity of each peak.

In order to provide six teams of students an opportunity to operate the equipment, the analysis must be completed within 30 minutes, not including counting time, for spectra containing fewer than 25 peaks. So that one team can set up an experiment while another team is analyzing data, the analysis of a spectrum must take place independently of the data acquisistion except for a short time, less than 2 minutes, for data transfer.

The system could be used by a student for thesis research, by a faculty member of sponsored research, or by a class researching an extensive laboratory project. All of these uses will require modifications to the programs which analyze spectra. As a result the system must be reprogramable. Minor reprograming might involve the writing of an executive program to execute functions in a specified sequence. Major reprograming would include rewriting the subprogram which performs a single function or adding a new function to the system.

The requirements stated above would be exceeded by an optimum system. The best possible system would respond within 5 seconds to any request for analysis, would process 200 peaks, and would perform many additional functions. These functions are discussed in Chapter III of this report; an example is the deconvolution of overlapping peaks. The minimum system should recognize overlapping peaks; a better system would separate them and provide the location and area of each. Still further improvement would resolve more than two overlapping peaks.

A subset of the requirements was established for a demonstration system. The reduced requirements enabled evaluation of the concept using already available resources. The basic functions of peak detection, peak location, peak integration, and linear energy calibration were selected for implementation, and a time limit of 5 minutes to perform these functions on one peak in a 4096 channel spectrum was established. Fifty peaks should be processed by the demonstration system within ten minutes. If a simple system could meet these requirements, then more complicated programs could be written to meet or exceed all of the requirements.

Scope

This study only considered one multichannel analyzer, the Series 2200 System Analyzer manufactured by Nuclear Data, Inc. To demonstrate feasibility, only a subset of the desirable functions of analysis were implemented; namely: data transfer, peak detection, peak location, peak integration, and the simplest of energy calibrations. Three manufacturers of multichannel analyzers were contacted to determine the characteristics and costs of current analyzers.

Assumptions

An assumption inherent in this study is that the multichannel analyzer performs as designed and will be maintainable over the expected lifetime of the microcomputer system. It is also assumed that the values contained in the channels are an accurate representation of the gamma ray spectrum.

Approach

The feasibility study was accomplished in stages. First, the electrical outputs available from the multichannel analyzer were determined. Then, a microcomputer was selected from those available and its capabilities studied. An interface between the multichannel analyzer and the microcomputer was designed, built, and tested. Next, software for the microcomputer was written, tested and refined. Several versions of the software went through this cycle, each version with additional capabilities. Finally, the software was timed in execution and evaluated in utility.

The determination of proposed system characteristics proceeded at the same time, primarily by comparing the capabilities of the current system against the requirements stated in the last section. In order to determine whether the proposed system was economical it was compared with three commercial systems of similar characteristics.

Sequence of Presentation

This report is organized in two major sections corresponding to the two purposes. The first section is a detailed description of the demonstration system and is divided into hardware, software, and evaluation subsections. The next section describes a proposed system by evaluating each of its characteristics and then compares that system with commercially available multichannel analyzers. A listing of each program in the demonstration system software is included in Appendix A. Appendix B describes testing of the interface board built for the demonstration system.

II. Demonstration System

The demonstration system consists of the hardware assembled and the software written to demonstrate a working, although limited, system for gamma analysis.

Hardware

The term "hardware" refers to the tangible pieces of equipment that make up the system. Hardware includes: the multichannel analyzer, the microcomputer, a memory board, a communications terminal, and an interface board which matches the signals of the multichannel analyzer to those of the microcomputer. These pieces of equipment will be discussed in order.

Multichannel Analyzer. (Ref 1) The multichannel analyzer used for the demonstration was a Series 2200 System Analyzer manufactured by Nuclear Data, Inc. The multichannel analyzer has three major modes of operation: acquire, display, and read-out. In acquire mode, the multichannel analyzer accepts pulses and increments the count in a non-volatile memory location, called a channel. The channel number incremented is proportional to the amplitude of the pulse. After a preset period of time, the analyzer terminates acquire mode and enters display mode. In this mode, the multichannel analyzer converts the counts stored in the channels to an analog signal to display on an oscilloscope the distribution of pulse heights as counts

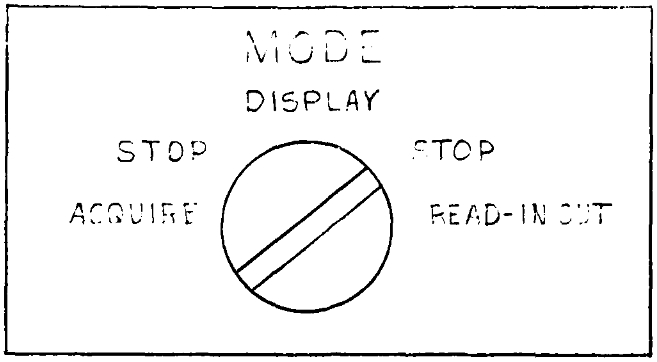

Fig 1. Multichannel Analyzer Mode Switch

versus channel number. This distribution, called the gamma ray spectrum, can be preserved by commanding the multichannel analyzer to enter read-out mode. The analyzer then steps through its channels in sequence sending the channel values and addresses to an auxiliary piece of equipment (teletype, magnetic tape drive, plotter, or microcomputer).

On the front panel of the analyzer are the operator controls. The operation of these controls will not be discussed except for the mode switch. The mode switch, illustrated in Figure 1, is used by the operator to place the analyzer in acquire, display, or read-out mode. There are two positions labeled STOP on the mode switch. The STOP position to the left of DISPLAY has the additional function of resetting the READ-IN/OUT mode. As a result, the switch must be left in the STOP position shown for all multichannel analyzer operations under control of the microcomputer.

On the back of the analyzer are three connectors; they are labeled PRINTER A, PRINTER B, and EXTERNAL. These co... .tors contain all the lines used to interface the multichannel analyzer with the microcomputer. The pin designation, name, and a brief statement of purpose for each line used in the interface are contained in Tables I, II, and III.

The multichannel analyzer was built in 1969 and contains logic elements (NAND and NOR gates) of the Diode Transistor Logic (DTL) family. The "high" or logic "1" of the DTL family is defined as +3 volts to +6 volts with respect to logic ground. The "low" or logic "0" is defined as -.5 volt to +.5 volt. These logical values are close to the nominal +5 volt "high" and 0 volt "low" of the Transistor Transistor Logic (TTL) family. Some TTL integrated circuits are specified to withstand more than +6 volts on their input lines, and the DTL circuits will treat +5 volts as a logic "1". Thus the two logic families can be interconnected if the proper TTL input devices are used.

In order to transfer data from the multichannel analyzer, the following sequence of signals is used. First, the EXT R line is held low. The analyzer enters read-out mode and sends the count in channel zero to lines MR1 through MR800K, sends the channel address (zero) to lines A1 through A4K and holds the 'I' line low. The coding of data on these lines is binary coded decimal (BCD). In other words, the channel data is first expressed as a 6 digit decimal number and then each decimal digit is coded as a 4 bit binary number. The 24 lines that result are known as MR1 (least significant bit) through MR800K (most significant

Table I

Multichannel Analyzer Signal Lines - PRINTER A Connector

Pin	Name	Direction	Purpose
A	MR1	Output	Channel Value - Ones Digit
B	MR2		
C	MR4		
D	MR8		
E	MR10	Output	Channel Value - Tens Digit
F	MR20		
H	MR40		
J	MR80		
K	MR100	Output	Channel Value - Hundreds Digit
L	MR200		
M	MR400		
N	MR800		
P	MR1K	Output	Channel Value - Thousands Digit
R	MR2K		
S	MR4K		
T	MR8K		
U	MR10K	Output	Channel Value - Ten Thousands Digit
V	MR20K		
W	MR40K		
X	MR80K		
Y	MR100K	Output	Channel Value - Hundred Thousands Digit
Z	MR200K		
a	MR400K		
b	MR800K		
d	GND	---	

(Ref 1:94-95)

Table II

Multichannel Analyzer Signal Lines - PRINTER B Connector

Pin	Name	Direction	Purpose
A	A1	Output	Channel Address - Ones Digit (Note 1)
B	A2		
C	A4		
D	A8		
E	A10	Output	Channel Address - Tens Digit (Note 1)
F	A20		
H	A40		
J	A80		
K	A100	Output	Channel Address - Hundreds Digit (Note 1)
L	A200		
M	A400		
N	A800		
P	A1K	Output	Channel Address - Thousands Digit (Note 1)
R	A2K		
S	A4K		
U	CMD	Output	Data Ready for Read Out (Note 2)
W	RDY2	Input	Finished With This Channel (Note 3)
e	I	Output	Channel Address Ends in Zero (When Low)
FF	GND	---	Logic Ground

Notes:

1. In Acquire mode, lines A1-A4K receive the count in channel zero, which is the number of seconds for which data has been accumulated.

2. Trailing edge (see text).

3. Rising edge (see text).

(Ref 1:84-86A)

Table III

Multichannel Analyzer Signal Lines - EXTERNAL Connector

Pin	Name	Direction	Purpose
B	EXT A	Input	Acquire Mode (When Low)
C	EXT R	Input	Read Out Mode (When Low)
d	GND	---	Logic Ground

(Ref 1:87-88A)

bit). Since the multichannel analyzer has 4096 channels, the channel address can be expressed using 4 decimal digits. The most significant digit needs only 3 binary bits so there is a total of 11 address lines (A1 through A4K).

After the data lines have been read, the RDY2 line is raised by the microcomputer. The multichannel analyzer raises the CMD line, advances to the next channel, changes the data lines (MR1-MR800K) and address lines (A1-A4K), and lowers the CMD line. The CMD line is high for approximately 10 microseconds. The RDY2 line can be lowered anytime after the CMD line goes high (less than 1 microsecond) and the output lines contain valid data any time after the CMD line returns low. The times above were experimentally verified.

Each succeeding channel is transferred by raising RDY2 briefly. When the last channel required has been transferred, the EXT R line is raised and the RDY2 line is raised to indicate that the microcomputer is finished with the last channel.

The microcomputer is one of several output devices. The other devices (teletype and magnetic tape drive) each have an interface unit which can be connected to the PRINTER B connector. Before the microcomputer interface was built, the teletype interface was connected to the magnetic tape interface which was connected to the multichannel analyzer. The connector on the back of the magnetic tape interface is also labeled PRINTER B. If the magnetic tape drive was in use, the teletype interface received no signals. When the microcomputer is added, the teletype interface is disconnected and the microcomputer interface is connected to the back of the magnetic tape interface. As long as the magnetic tape unit is not being used, the microcomputer interface receives the same signals as if it were connected directly to the multichannel analyzer. In this manner, the tape drive can be used as before and the teletype function is one of those performed by the microcomputer.

Microcomputer. The procurement cycle for data processing equipment is longer than the time available for this study. Therefore, computers on hand and available were surveyed, and one selected which appeared to be suitable for a demonstration of feasibility. The microcomputer used was a Single Board Computer, model SBC 80/20 manufacted by Intel Corporation (Ref 2). The single board computer contains an Intel 8080 microprocessor and associated support devices in a 4 slot card cage in the Intel Multibus configuration. Memory available on the microcomputer board includes 4 sockets for type 2708 Erasable Reprogrammable Read Only

Memories (EPROM) providing 4096 bytes of program storage. Also available is 2048 bytes of dynamic Random Access Memory (RAM) for data storage. Three timer/counters are available in an Intel 8253 integrated circuit; one timer is used to produce a baud rate clock signal for the serial input/output (I/O) controller. Serial I/O is performed by an Intel 8251 Universal Synchronous Asynchronous Receiver Transmitter (USART) which produces standard serial interface signals (RS232C). An interupt controller is available but is only used to regain control of the system when a program malfunctions.

The most important feature of the single board computer is its parallel input/output capability. Six ports, each 8 bits wide, are available. Three input ports are used to receive signals on 24 lines and one output port is used for 8 control lines. One of the remaining ports was used to display results during interface testing (see Appendix B). These input and output lines are available at the edge of the microcomputer board and are led from there to the interface board by printed circuit edge connectors and flat wire cable. The input and output ports were configured by wire-wrap jumper connections on the microcomputer board. An input port samples the signal lines at the time a programmed 'IN' instruction is executed. The output lines are given new values by an 'OUT' instruction which they hold until the next 'OUT' to that port. The logic levels used by the parallel ports are the TTL levels mentioned earlier, however, the maximum input voltage is 5.5 volts. Therefore, some buffer device is required between the DTL output of the multichannel analyzer and the TTL input of the microcomputer parallel port.

Interface Board. Buffering, input routing, and interupt control are performed by devices on the interface board. Input routing is necessary because there are 41 output lines from the multichannel analyzer and 24 input lines to the single board computer. One reset and two interupt buttons on the interface board provide control over the single board computer. A one-digit display connected to 4 bits of a spare output port provides a status monitor which was used in interface testing. An additional function of the interface board is to hold the cable connectors for cables leading to the multichannel analyzer.

These functions are diagrammed in Figure 2, which depicts signal flow in a block diagram. The inverters are used as buffers to convert DTL levels to TTL. The box labeled MUX connects one set of 24 input lines or the other to the three 8-bit input ports. Which set is selected depends on the logic level of the select line. The debounce circuits allow only one interupt per push of the button even though the switch contacts may make multiple contacts. The box marked LED is a light-emitting-diode display of a digit 0 to 9 or a letter A to F depending on the hexadecimal value of the 4 output lines.

The TTL logic devices were selected not only for the function they performed but also because they were readily available. The 74LS04 inverter was chosen because its input

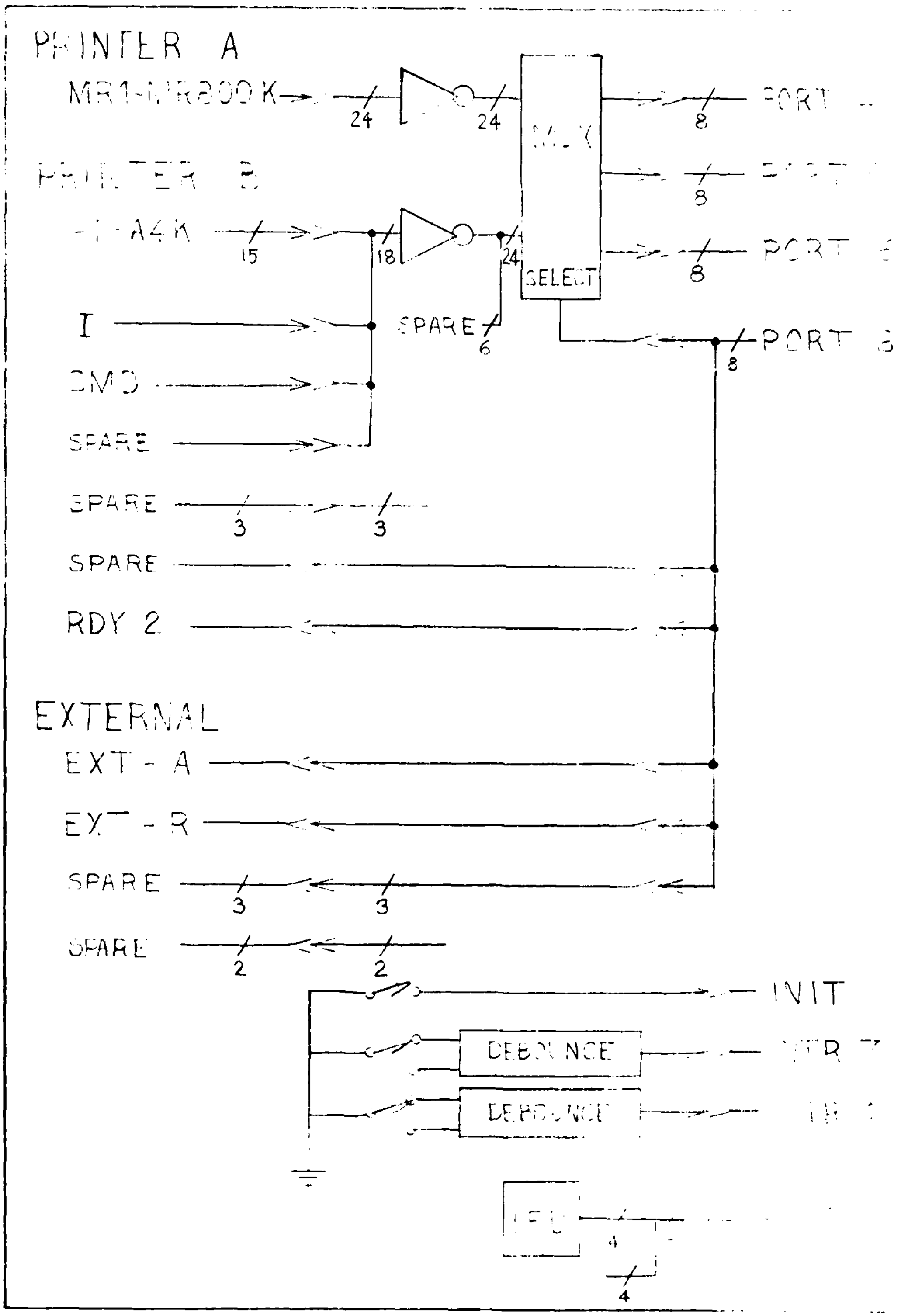

Fig 2. Interface Board Block Diagram

voltage specification is +7 volts maximum. Each integrated circuit contains 6 inverters so 7 circuits are required to buffer the 41 DTL lines from the multichannel analyzer. After buffering through the inverters, the signals are applied to type 74157 2-line to 1-line data selectors/multiplexers. With 4 output lines per integrated circuit, 6 circuits are required for 24 lines input to the microcomputer. Two of the circuits were changed to type 74158 devices (which inverts its outputs) because one of the input ports inverts its signals and the other two input ports do not invert their signals. The devices and assorted hardware were laid out as in Figure 3 on the interface card which plugs into one slot of the card cage. The devices are connected together with wire-wrap as depicted in the schematics of figures 4 and 5.

Random Access Memory (RAM) Board. Also plugged into the card cage is an Intel SBC016 board providing 16,384 bytes of additional data storage. The additional RAM is used for three purposes. The 4096 bytes of program are moved from EPROM to RAM before execution. In this manner, changes can be made in a program under development without having to erase and reprogram the EPROMS. Approximately 1000 bytes of RAM are used for COMMON data storage (see software section for details). The remaining bytes are used to store about 2800 channels of multichannel analyzer data. While the present board is sufficient for system testing, a 32,768 byte RAM board is required to test and time the full 4096 channel capability of the multichannel

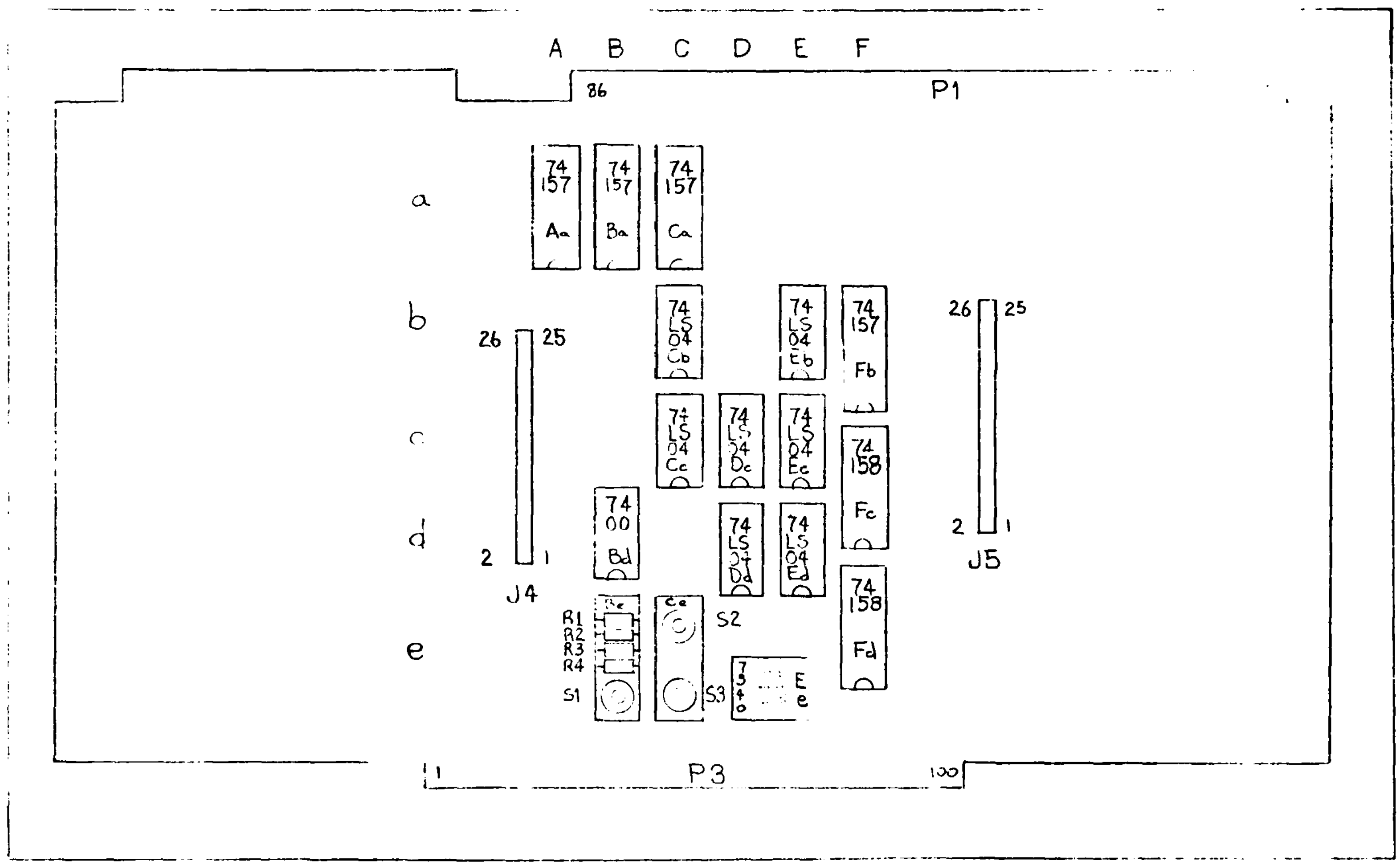

Fig 3. Interface Board Layout

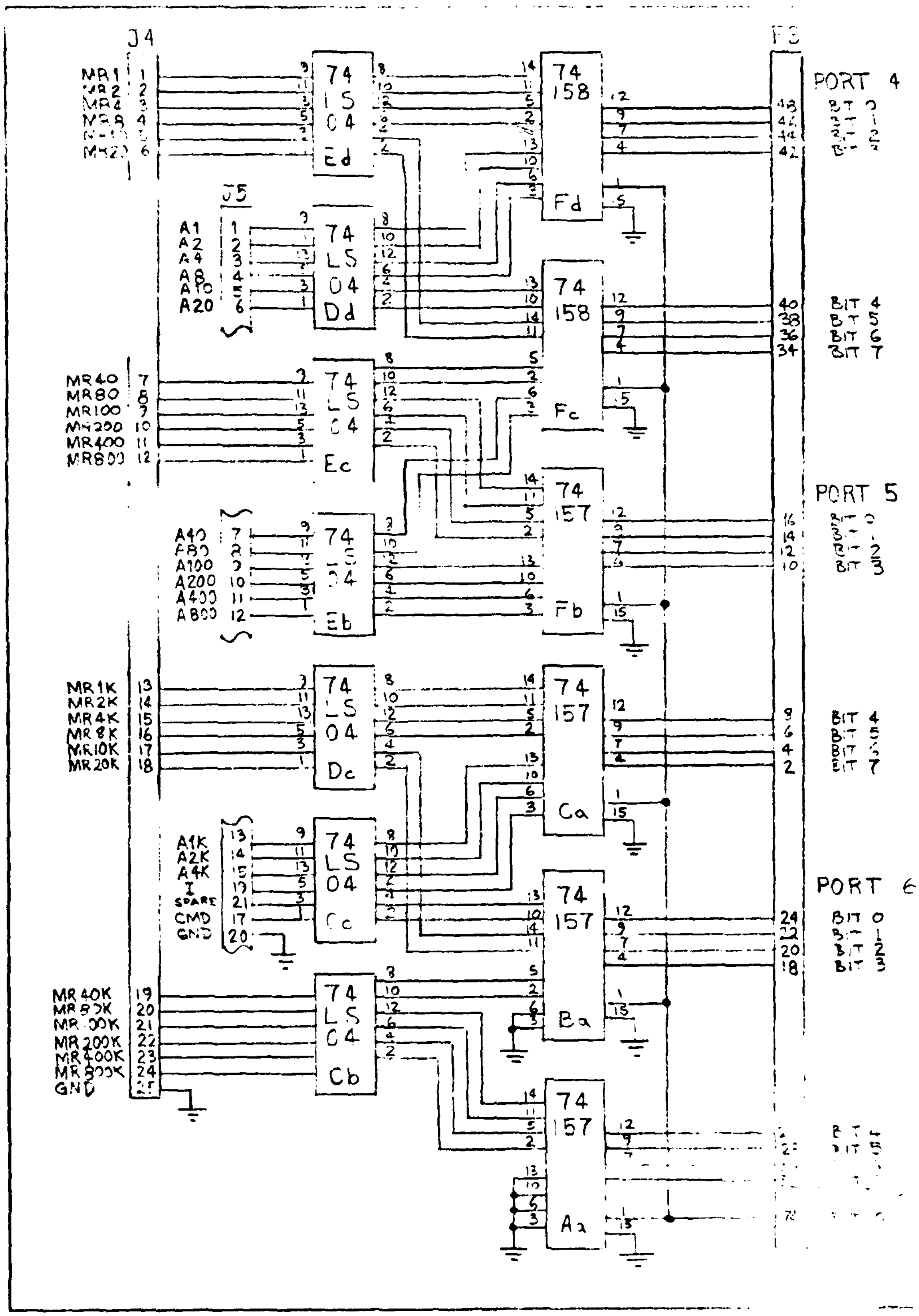

Fig 4. Interface Board Input Schematic

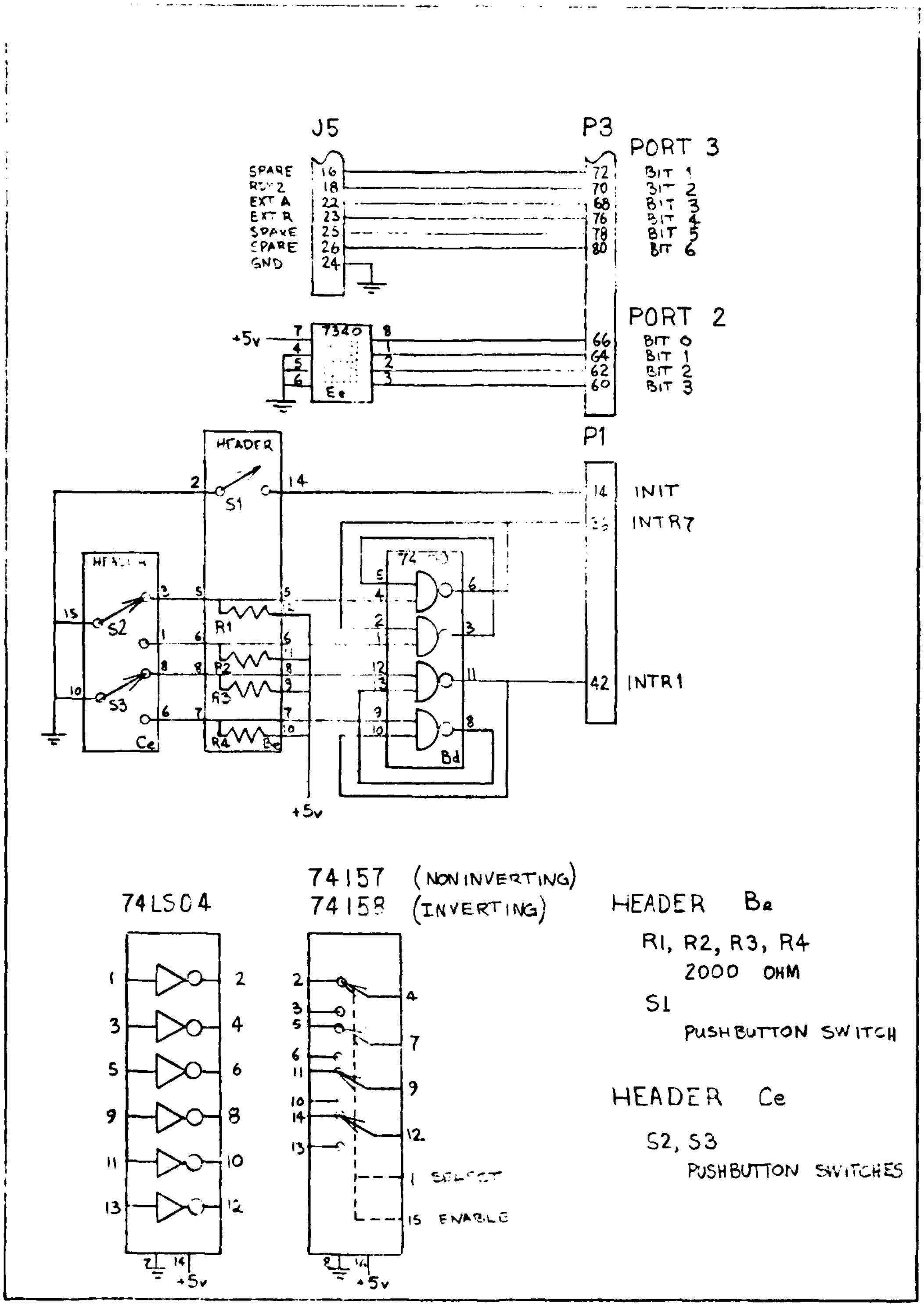

Fig 5. Interface Board Output, Interupt, and Devices

an... ...er.

Terminal. The final hardware item required is a communications terminal to send commands to and receive output from the microcomputer. Again, availability was the driving factor in selecting the terminal to be used. The terminal is a CDC communications terminal modified locally to interface with the standard serial protocol (RS232C). The terminal operates at 300 baud (bits per second, 30 characters per second) and provides keyboard input and printed output. The terminal connects with a flat cable and edge connector to the front of the single board computer.

Hardware Summary. From available hardware, a microcomputer system was assembled. Each of the components (except the terminal) plugs into the card cage. The card cage itself is mounted on a chassis with a power supply for each of the voltages required (+5, -5, +12, -12 volts DC). The card cage and power supplies were salvaged from a previous thesis project (Ref 3). After the system was assembled and the interface board designed and built, the interface was bench tested. Test procedures and results are described in Appendix B.

Software

Because no mass storage device (disk, magnetic tape, paper tape) was available, software for the demonstration system was developed on another system. This system was the Intellec Series II Model 210, with two 8-inch floppy disk drives, manufactured by Intel Corporation. Software was developed under the ISIS-II (V4.0) operating system (Ref 4). Once the software programs had been developed and tested on the development system, the programs were "burned" into Erasable Programmable Read Only Memory (EPROM) in the form of absolute machine code. The EPROMS were then inserted into sockets on the single board computer and the programs were executed on the microcomputer.

Functions. The programs are used to perform the various functions of gamma ray spectroscopy. Many functions are desirable; but, due to limitations in the amount of EPROM storage and time available for this research, only the most important could be implemented. The first function implemented must be the transfer of data from the channels of the multichannel analyzer to the memory of the microcomputer. The spectrum in memory can then be scanned for peaks. When a peak is detected, its limits are found, background is determined and subtracted from a sum of counts under the peak to obtain net peak area. At the same time the peak location is determined. These three functions are called peak detection, net peak area determination, and peak location. The final basic function implemented was energy calibration.

Source Languages. The functions described above were implemented by programs written in either 8080 Assembly Language with Intel opcodes (Ref 5) or FORTRAN (Ref 6). FORTRAN is the language of choice because it is a high level language, a compiler for it is available, and nuclear engineers, who are expected to continue this project, are fa...iar with it. For functions that cannot be performed in FORTRAN or are unwieldy or time and space consuming, assembly language coding was used.

Modules. The particular reason for coding in a given language will be stated for each of the modules of the demonstration system software. Also discussed for each module will be the purpose, algorithm and limitations of the module and its sub-modules. Compilation/assembly listings of each module are located in Appendix A. The structure of the programs written for this study is shown in Table IV; the modules will be discussed in that order.

The MONIT module is the first module executed. It performs the functions of an operating system:

- Copy programs from EPROM to RAM for execution.

- Initialize the serial input/output channel, baud rate timer, parallel input/output ports, and interupt controller.

- Accept commands to display memory contents (D), change memory contents (S), or start program execution at a particular memory location (G).

- Provide the lowest level serial input and output routines.

The MONIT module is written in Assembly Language because of its low-level input/output and machine-control functions. The subroutines contained in MONIT are CI and CO. CI reads one character from the terminal keyboard and CO prints one character on the terminal printer. MONIT provides the minimum number of commands to allow program debugging. A far more sophisticated monitor is supplied with the single

Table IV

Demonstration System Program Structure

Level 1	2	3	4	Purpose
MONIT				Initialize and perform debug functions
	GAMMA			Call subroutines in sequence
		GETDAT		Get data from multichannel analyzer
		PKSRCH		Scan spectrum to detect peaks
			PKPROC	Process each peak for location and area
		ENCAL		Perform energy calibration
Support Routines				
INT4				Perform 32-bit integer arithmetic
ASSEM				Provide formatted input and output

board computer but that monitor occupies 2048 of the 4096 available bytes of program storage. MONIT only uses 502 bytes. The default address for the G command (begin program execution) is the first location of the GAMMA module.

The GAMMA module's only function is to call its subordinate routines in sequence. It is written in assembly language because it originally performed the parallel port initialization and also because it is shorter.

GETDAT is the first module called by GAMMA, and is also written in assembly language for two reasons. GETDAT performs the manipulation of the parallel ports which causes the multichannel analyzer to sequence through its channels and send their counts across the interface. Bit-oriented input/output is the primary reason for coding in assembly language. The channel counts are then converted from 6 Binary Coded Decimal (BCD) digits to a 32-bit binary integer representation and stored in memory. GETDAT asks the user how many channels to transfer. An answer of zero implies that previous data is to be used. The maximum number of channels transferred is 4096, so a small savings in the time to transfer one channel can make a big difference in the time required to complete a full transfer. Saving time is the second reason to code in assembly language.

PKSRCH, however, is written in FORTRAN because it involves a number of arithmetic calculations which are much easier to express in the algebraic syntax of FORTRAN. The name PKSRCH is a contraction of "Peak Search", which expresses the purpose of this module. The spectrum is scanned, and each peak which meets the detection criteria is identified. The algorithm used is from a program named GAMMA (Ref 7), which is currently used at the Air Force Institute of Technology to reduce gamma-ray spectra on the large mainframe Cyber series computers. The current program was originally written by David Guice (Ref 8) who obtained the peak search algorithm from a published paper (Ref 9). A measure of curvature is computed at each point of the spectrum and compared with a background value (50 channel forward average). If the ratio exceeds a user specified criteria, then a peak has been detected. The PKSRCH routine determines the first and last channel at which the criteria are exceeded and passes the channel numbers to the PKPROC module. To reduce the number of EPROM bytes occupied by the PKSRCH routine, the algorithm was limited to a fixed peak width. This peak width is defined by its Full Width at Half Maximum (FWHM) at 3 channels. This value is appropriate for most of the gamma-ray spectroscopy accomplished with this multichannel analyzer (Germanium-Lithium Drifted detector with 4096 channels spanning an energy of approximately 2 Mev). Future implementations which do not have a program space limitation should make other peak widths available as an option. In summary, PKSRCH performs the peak detection function by finding the sharp change of slope at a peak.

PKPROC performs the other peak functions: locate and compute area. With its many arithmetic calculations, PKPROC is a natural for, and was originally written in, the FORTRAN Language. However, because it was the largest module in the entire program it was the best candidate for recoding in assembly language when the entire program was too large to implement. The FORTRAN source code has also been included in Appendix A and should be used for any future implementation. The name PKPROC is a contraction of "Peak Process". Processing is accomplished in several steps which are illustrated in Figure 6.

- The channels between the curvature limits supplied by PKSRCH are scanned for the channel with the maximum number of counts.

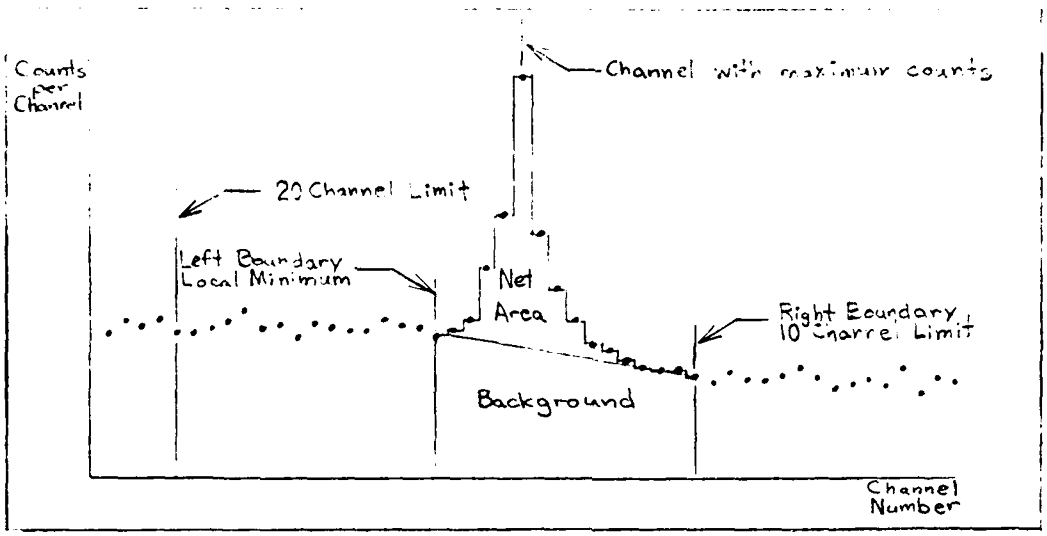

Fig 6. Processing a Typical Peak

- From the maximum channel, a scan is made to the left to find a local minimum. In other words, the scan is continued as long as the channel counts are monotonically decreasing. A limit of 20 channels is placed on the scan.

- A scan is made to the right from the maximum channel; also searching for a local minimum. The scan is terminated at 10 channels even if the counts are still decreasing.

- The left and right minima are defined as the peak boundaries and the channel counts at those points are assumed to be the background value at those points.

- A trapezoidal background is assumed by constructing a straight line between left and right minima.

$$\text{Slope} = \frac{\text{Counts(R)} - \text{Counts(L)}}{\text{R} - \text{L}} \qquad (1)$$

Where,

L = Channel Number of Left Bound
R = Channel Number of Right Bound

Counts(i) = Counts in Channel i

- Three sums are computed during a scan from the left to the right boundary. One sum is of total counts in each channel, another is of net counts after subtracting background calculated for that channel, and the last is of net counts times channel number.

$$\text{Total} = \sum_{i=L}^{R} \text{Counts}(i) \qquad (2)$$

$$\text{Net}(i) = \text{Counts}(i) - \text{Slope} \times (i-L) - \text{Counts}(L) \qquad (3)$$

$$\text{Area} = \sum_{i=L}^{R} \text{Net}(i) \qquad (4)$$

$$\text{Firstm} = \sum_{i=L}^{R} \text{Net}(i) \times i \qquad (5)$$

The third sum, when divided by the second sum is the first moment of the peak.

$$\text{Centroid} = \text{Firstm} / \text{Area} \qquad (6)$$

If the peak is an approximation of a gaussian distribution (tailing effects are ignored), then the first moment of the peak is the best estimate of its centroid (Ref 10:68-69).

As PKPROC performs these steps it prints the following information about the peak:

- sequential number to identify the peak,
- curvature limits passed by PKSRCH,
- channel with maximum counts,
- whether limits were encuntered in searching for peak boundaries,
- boundaries used,
- total counts,
- net counts, and
- peak location to one tenth channel.

This algorithm is rather limited. It does not recognize doublet peaks, uses only two channels to determine background at the peak location, and will use a one-channel minimum to define peak boundary. Any follow-on program not space limited should at least implement the full algorithm contained in program GAMMA (Ref 7). A further limitation is the assumption of a 30 channel maximum peak width, which is acceptable for the present use but should be made a user specified parameter.

ENCAL is also limited in that it performs an energy calibration based on only two peaks, assumes linear energy dependence on channel number, and requires that the spectrum calibrated contain the calibration peaks. Energy calibration is performed by obtaining peak number and energy from the user for two peaks, calculating the slope and offset, and calibrating each peak in energy by the linear equation:

$$\text{Energy} = \text{Offset} + \text{Slope} \times \text{Channel} \tag{7}$$

ENCAL is coded in FORTRAN because it performs arithmetic calculations. Energy calibration is the last of the gamma ray spectroscopy functions which has been implemented on this demonstration system. The next two modules contain the common subroutines called by several of the previous modules.

The INT4 module contains 15 subroutines which perform 32-bit integer arithmetic. These routines were written because the floating point subroutines which are part of the FORTRAN package take up too much program space. An additional benefit of integer routines is that they operate faster. The FORTRAN integer routines could not be used because they only permit integer values between -32768 and 32767. The maximum counts in a channel is 999999. In order to be able to use FORTRAN coded routines, it was decided to use real variables to hold those values in excess of the 16-bit limits. The FORTRAN floating point routines were replaced by 32-bit integer routines of the same name. In some cases, variables with non-integer values were required. The slope of the background line used in PKPROC is a good example. To handle these cases, scaled integers were used. A scale factor, that was ten to a power, was chosen for each variable. The value stored and manipulated was the integer nearest to the actual value times the scale factor. The manipulation routines are listed in Table V with their name and purpose. Most of the routines take two arguments. The

Table V

32-Bit Integer Arithmetic Routines of Module INT4

Name	Purpose
FMUL10	Multiply by 10
FQFABS	Absolute value
FQFAD4	Add
FQFCLR	Clear accumulator
FQFCM4	Compare
FQFDV4	Divide
FQFERH	Report error
FQFFLM	Same as FQFLD4
FQFLD4	Load accumulator
FQFML4	Multiply
FQFMP4	Same as FQFML4
FQFNEG	Negate
FQFSB4	Subtract
FQFSET	Same as FQFCLR (See note)
FQFST4	Store accumulator

Note: Calling sequence is different for FQFSET

first argument is a 12 byte area in memory which is logically divided into three 4-byte registers named ACC, EXT, and AUX. The ACC register is the place where one operand is stored on entry and the result is stored on exit. The EXT register is an extension of the ACC register and is used during multiply and divide operations. The AUX register is used during multiply and divide operations to hold the absolute value of the second operand, if that operand was originally negative. A significant feature of these routines is that the number of bytes for each operand is specified by a parameter. In other words, to produce 24-bit or 64-bit integer arithmetic routines it would only be necessary to change one character of the source and r:- --mble the routines. No comparable routines were found in the literature or were known to the faculty of the Electrical Engineering Department of the Air Force Institute of Technology. The closest routines that could be found were floating point arithmetic routines (Ref 11). The general algorithms used were from Flores (Ref 12). In summary, the INT4 module contains 32-bit integer arithmetic subroutines which replace the FORTRAN floating point arithmetic subroutines.

The ASSEM module contains six subroutines written in assembly language for compactness. All but one subroutine can be called from a FORTRAN program and together they replace the FORTRAN input output package. The subroutines are individually discussed below.

CRLF defines the end of a printed line by sending the carriage return and linefeed characters to the communications terminal. After each of the characters, a special character called a synchronous idle is sent. The idle is not printed and is only sent because the particular terminal used with this system under some circumstances will not process the character following a carriage return. No routine in this program sends carriage return or line feed characters except CRLF.

ININT obtains an integer reply from the user. One of the other subroutines is used to print a question which can be answered with a non-negative integer less than 32,767. The user indicates his reply by typing the digits followed by a carriage return on the terminal keyboard. ININT is called with two arguments. The first is an integer variable which will be assigned the reply value, and the second is a default value which will be assigned to the variable if the user types carriage return without any digits. No edit capability (such as backspace) is provided, but a value too large or an invalid character will cause the message "INVALID, TRY AGAIN" to be printed, and the user can then enter the correct value.

OUTCHR can be used to ask the question for which ININT receives the reply. OUTCHR is called with one argument, a character string between single quotation marks ('). The string is printed by the communications terminal.

OUTSTR performs the same function as OUTCHR but, in assembly language, is easier to call than OUTCHR. OUTSTR is the one subroutine in the ASSEM module which cannot be called from a FORTRAN coded routine.

OUTINT is used to print integer results. It is called with two arguments. The first is an integer variable whose value should be non-negative. The value is printed as a string of decimal digits. If the number of digits is less than the value of the second argument, then enough spaces will be printed before the string of digits so that the entire string will be as long as the second argument value. The purpose of the second argument is to provide a means for printing tables of numbers with columns whose right edge line up vertically. The construct is similar to FORTRAN formatted output.

OUTFLT also produces formatted output. The first of two arguments is a real variable which contains a 32-bit signed integer which is scaled. (Real variables with 32-bit integer values were explained above in the paragraph on the INT4 module.) The second argument serves two functions. The first function is to provide the scale of the integer; that is, the location of the decimal point. A scale of zero means there is no decimal point; larger values indicate the number of digits after the decimal point. The second function of the second argument is to specify the minimum length of the string of output. It is used the same way as the second argument of the OUTINT subroutine. The value of the second argument of the OUTFLT subroutine is SCALE x 100 + LENGTH. This encoding scheme limits the minimum length parameter to 99, but that is long enough for any contemplated use. The output produced by the OUTFLT routine is:

- a minus sign if the value is negative,
- at least one digit even if zero,
- a decimal point if scale is greater than zero,
- a string of digits whose length equals scale.
- If the length of the string of characters is less than the minimum length specified in the second parameter, then spaces are inserted before the string to make its length equal the minimum.

Together, these six subroutines provide an extensive input/output capability to the FORTRAN and assembly language modules.

Software Summary. The modules are assembled or compiled separately and linked together as a single program with absolute addresses. The absolute program is then written into 4 EPROMs, and the EPROMs are installed in the single board computer. The single board computer, the random access memory board, and the interface board are inserted in the card cage, and cables are connected between the interface card and the multichannel analyzer. At this point, the demonstration system is ready to be tested.

Evaluation of the Demonstration System

The purpose of the demonstration system was to determine whether a microcomputer connected to the Nuclear Data Series 2200 System Analyzer could be used for analysis of gamma-ray spectra. The determination requires answers to the questions:

- Does it work?
- How long does it take?
- Is there room left for additional functions?

Performance. The complete demonstration system has not been successfully operated. The transfer of data from the multichannel analyzer has been successful. The other analyzer functions have been executed on the software development system and produced the expected results on limited test data. The only portion of the program which is known to be not working is the first portion of the MONIT module which copies the program from EPROM to RAM. Based on extrapolation from these separate tests it is possible to state that the system will work when the remaining minor problems are resolved. The limitations mentioned in the software section reduce the usefulness of the system, but they are not limitations of the approach but only the method of program storage. But even with the limitations, the following capabilities have been added to the multi-channel analyzer: peak detection, peak location, net area computation, and energy calibration.

Timing. Each of these functions has been timed. In the case of peak location, net area computation, and energy calibration the computation for each peak takes less time than printing the result for that peak. Therefore, the time required for these functions is determined by the output device used. Even with the 300 baud (30 characters per second) printer used, data was printed at 6 seconds per peak or 5 minutes for 50 peaks. For the remaining functions, reveals the program execution speed. Transfer of 4096 channels from the multichannel analyzer to the microcomputer took 54 seconds. A scan for peaks of 4096 channels took 2 minutes 58 seconds. In both cases, the times are limited by the time required to perform a multiplication or division of 32-bit integers. These times are tolerable; they imply a complete spectrum analysis in 4 minutes for one peak and 6 minutes for 50 peaks. Additional functions will add to this time. As an extreme example, data smoothing is expected to take between 2 and 4 minutes for 4096 channels. These results represent a significant improvement over the present alternative of 1 to 3 days.

Memory. Additional functions will also take more memory, so it is imperative to examine how much memory space has been used by the current system. The uses for memory can be divided into essentially 3 areas: program storage, temporary data storage, and common data storage. Program storage has been limited by the available system to 4096 bytes. It is contemplated that program storage would grow to fill any possible allocation as additional functions were added. Temporary data storage refers to temporary variables maintained by various subroutines whose values are not preserved between calls to that subroutine. Stack space falls in this category. Space required for temporary data is less than 432 bytes. The stack has never overflowed, but it is not known how much smaller it could be. Common data storage includes 20 bytes per peak analyzed, 4 bytes per channel stored, and 36 bytes general overhead. Currently 4096 channels and 50 peaks are allocated for a total of 17420 bytes. Storage generally comes in 16384 byte increments, so 32768 bytes of storage are required to analyze 4096 channel spectra. Based on the limitations imposed on the demonstration system by the 4096 byte limit on program storage, it is also clear that any future system will need more memory devoted to program storage.

Summary

Using available equipment, a microcomputer system has been assembled; and an interface to the multichannel analyzer has been designed, built, and tested. Software for the demonstration system has been written, compiled, and

integrated, but only partially tested. Based on the partial test results and timing of the incomplete system, the concept has been proven feasible and definite indications have been found as to what shape the next system should take.

III. Proposed System

This section describes characteristics proposed for inclusion in a system to be developed specifically to enhance gamma ray spectroscopy on the existing multichannel analyzer. The first portion describes the hardware that should be purchased, the second portion identifies the software that should be written, and the final portion discusses hardware cost as compared to current multichannel analyzers which have enhanced capabilities.

Hardware

The discussion will not identify specific pieces of equipment by manufacturer and part number, but rather will indicate the desirable and mandatory characteristics the equipment should exhibit.

Microcomputer. The first equipment item selected should be the microcomputer, because the other equipment must be compatible with the microcomputer. The system selected should be available "off the shelf" to avoid long delays while obtaining parts or designing, building, and correcting a new system. Existing systems also have many compatible peripherals available. The microcomputer should have an accessible bus structure with a number of slots available. Spare slots should be available after all the basic equipment is attached so that additional capabilities can be added later. A major division exists between 8 bit and 16 bit microcomputers. Very briefly, 16 bit machines are faster and more expensive. The demonstration system was an 8 bit machine, and its speed was shown to be adequate for gamma ray spectroscopy. In addition, software developed for the demonstration system can be used, in part, on a new system if the new microcomputer is based on the Intel 8080 or Zilog Z-80 microprocessor. So, unless other considerations (such as availability through the procurement system) warrant a change, an 8 bit microcomputer is recommended.

Memory. The microcomputer will determine the form of memory, but it can be stated that 32,768 eight-bit bytes is the minimum required to analyze spectra from a 4096 channel analyzer. Additional memory will increase the amount of program available at one time and will therefore increase processing speed. Additional memory will also make possible some additional application programs and make it easier to use programs originally developed for larger machines. At least 65536 bytes of memory should be purchased.

Disk Drives. A mandatory peripheral for the next system is one or more disk drives. A disk drive will make it possible:

- to develop software on the system,
- to run overlayed programs which are larger than the memory size,
- to save gamma ray spectra for comparison with others or for archives, and

- to maintain a library of nuclides and their gamma ray signatures.

A lack of a bulk storage medium was the single factor which most limited the demonstration system.

Once the decision to buy a disk drive is made, the option to buy more than one should be examined. There are many times when two disk drives are very useful, the primary being when the information on one diskette is being copied to another. This operation occurs rather frequently to back up the information in case the original becomes unreadable. The alternative is to read a portion into memory, change disks in the one drive available and write the portion on the second disk. The disks are changed again and the operation repeated several times until all the information is transferred. Not only is this a tedious process but also prone to error which may result in the destruction of the very information which was being preserved.

Many disk drives are available, but the basic choice is between a 5 inch floppy disk, an 8 inch floppy disk, and a hard disk. The trade off is between storage capacity and cost. The hard disk costs the most, stores the most, and is the most reliable. However, for archival storage, many disks would be required, and hard disks are extremely expensive compared to floppy disks. The 8 inch floppy is recommended because each disk stores five times as much information as the 5 inch floppy disk but the disk drive only costs twice as much and the disks themselves are about the same price. In addition, the 8 inch floppy is the most common drive for microcomputer systems.

High Speed Math Unit. Many microprocessor systems have a hardware device available which provides arithmetic operations on 32-bit floating point operands at high speed. Since many arithmetic operations are performed, the time required to analyze a gamma-ray spectrum is directly dependent on the speed of the arithmetic operations. A high speed math unit would increase the speed of analysis and also reduce the amount of memory taken up by programs. While highly desirable, the high speed math unit is not absolutely necessary. The microcomputer purchased should, however have the capability to add the unit later when new programs are added which would be useful if only they operated faster.

Terminal. The final piece of hardware for the proposed system is a communications terminal. Through this terminal the user will communicate with the microcomputer. Because of the conflicting requirements of hard copy and spectrum display, the terminal will probably be more than one piece of equipment. At the very least the terminal will consist of a keyboard and a printer. The keyboard is necessary to type in programs during software development and will probably be used to select modes of operation and other choices during spectrum analysis. The printer is necessary to record results of analysis and program listings.

Another option for a communications terminal would be a cathode ray tube (CRT) terminal with a keyboard and screen with a separate printer. It is easier to enter programs in de- -pment and display intermediate results in operation with a CRT terminal. If the terminal has a graphics capability, then a spectrum display could be generated on the terminal. Some form of spectrum display is essential and, while the display currently provided by the multichannel analyzer is adequate, a microcomputer driven display would be much more versatile.

An alternate display option would be to connect a digital to analog converter between the microcomputer and an oscilloscope. This scheme has the advantage of being cheap and easy to implement in hardware, but requires software development, a large fraction of the microcomputer processing time while the display is running, and a sizable amount of memory. A separate processor with its own memory could be built, but this is just the description of a graphics terminal.

The terminal could also have exotic methods of user input in addition to the keyboard. A paddle, joystick, or light pen in conjunction with the spectrum display would make it easier to specify particular regions or peaks in the spectrum during analysis. With a menu of various input options or modes of operation displayed, any one of the alternate inputs would make selections easy.

Many directions are available in the choice of a communications terminal. The basic needs which must be satisfied are a keyboard and some form of hard copy output. Spectrum display is highly desirable and analog input is nice to have. Many options are also available in the rest of the system. An 8 bit microcomputer with at least two open slots, as much memory as possible, and two 8 inch floppy disk drives are recommended.

Software

All the hardware mentioned above is not useful without software. This section will discuss some of the general characteristics that the software for analysis of gamma-ray spectra should have and will then discuss some of the analysis functions which could be implemented. The various functions are organized in major groupings to gather similar functions together.

General Considerations. A primary requirement for all software currently being written is that it be structured. Structure in this case implies that tasks to be performed are analyzed before the program is written and broken down into small units, each of which has a simple, well-defined role in the overall system. Structure makes concentration on a single module or submodule possible without the programmer being required to remember what all of the other modules are doing. A well-defined structure also allows more than one person to work on a program and allows changes to be easily made.

Almost as important as structured design, is the selection of a high level language in which to program. A high level language (such as FORTRAN, Pascal, or Ada) allows the programmer to think on a higher level and depend on the compiler to keep track of details. The FORTRAN-66 language

(Ref 13) is not suited to structured programming within each routine. The FORTRAN-77 language (Ref 14) has numerous additions, most notably the IF-THEN-ELSE control construct, which allows structured programming, and the CHARACTER data type which allows direct machine-independent manipulation of character strings. The FORTRAN language is also nearly a standard language for Nuclear Engineering, and almost all people who would use the program for gamma-ray spectroscopy would understand that language. While a high level language is easy to understand and to use for writing programs, it is not the most efficient in either time or memory requirements. With a disk drive and 65536 bytes of memory, memory use is not a factor but, in some situations, execution speed will be. For subroutines which are used very frequently inside the innermost loop of a program, assembly language coding will yield significant improvement in the time required to execute the program. Another likely use of assembly language is to program the input/output operations with the multichannel analyzer interface. These operations would occur frequently at the innermost loop for the transfer of data from the multichannel analyzer to the microcomputer.

Data transfer is only one of many functions to be performed on this system. The next function to be performed is selected in a fixed order in the demonstration system (and program GAMMA, of Ref 7). The fixed sequential order of functions is suitable when only a few functions are implemented. However, the system proposed here will have functions and they should be organized in a command structure. The user would select the next function to be performed from a set of available commands. Certain sequences could be prohibited or simply not available; for example, energy calibration cannot precede peak location which cannot precede peak detection. However, peak location could be performed twice (with different parameters) after peak detection.

Programming a sequence of functions should also be possible. Experiments often require large amounts of repetitive processing and repeated tasks are better suited to a computer than a human. So, the user should be able to write a supervisory program which will execute analyzer functions in sequence, specify parameters and even make decisions based on results. The user should also be able to temporarily replace one or more program modules to tailor the data analysis to his particular experiment.

In general, the software for the proposed system should be highly structured in a high level language, preferably FORTRAN-77, with some input/output or inner loop subroutines coded in assembly language. The various functions should be commanded in the sequence desired either by the user at the communications terminal or by a user written supervisory program. The user should also be able to modify functions for a particular purpose.

<u>Transfer Functions</u>. Before data is available to analyze, it must be read into the microcomputer memory. Moving data to and from the microcomputer memory is the

purpose of the transfer function. One source of the data is the multichannel analyzer, which is the original source of all gamma-ray spectra. Another source is the disk drive. Three transfer functions should be available on command to transfer data:

- from the multichannel analyzer to the microcomputer memory,
- from the microcomputer memory to the disk drive, or
- from the disk drive to the microcomputer memory.

The disk transfer operations should allow the specification of a name for a particular set of data. The data transfered should include not only channel counts but also the results of any analysis already performed. The multichannel analyzer transfer function should allow any contiguous set of channel counts to be transferred (ignore the contents of the channels before the desired start channel).

Manipulation Functions. Once the data is in memory, there should be several manipulation functions available. One of the most common functions desired is data smoothing. Bevington (Ref 10:255-260) gives the arguments and cautions for data smoothing; he also provides an algorithm for smoothing random data which contains gaussian peaks.

Another useful function would be a shift and stretch routine to move the entire spectrum to the left or right by a specified number of channels and at the same time stretch or shrink it from the current number of channels to a specified number of channels. With this function, it would be possible to compare two spectra which were not taken at the same equipment settings (different gain or offset). Interpolation would be necessary when stretching and averaging when shrinking.

The comparison of two spectra would be accomplished by a strip function, which subtracts a specified multiple of one spectrum from another spectrum. For example, a background spectrum can be subtracted from a sample spectrum. The two spectra need not be memory resident. The first spectrum, from which the second is subtracted, will be in memory, and the results will be in memory; but the second spectrum, whose ratio is subtracted, could be a disk file.

Peak Processing Functions. After data manipulations are complete, information about the peaks will be required. The peak processing functions (detect, locate, integrate) will probably come as a package. However, peak location could be followed by an edit cycle to delete those peaks about which further information is not required, and to add those peaks that the algorithm missed. Improvements to the demonstration system peak processing algorithms are certainly in order and will include allowing different peak widths, more channels in peak edge detection, and more channels in the background average. A further improvement would be the addition of a peak fitting algorithm which would do a better job of peak location and deconvolution of overlapping multiple peaks (Ref 15: 735-739). Another function would compute the ratio of peak areas. Either the ratio of one peak to others in the same spectrum or the ratios of corresponding peaks in two different spectra. A ratio determined by this function could be used as the ratio for the strip function.

Calibration Functions. For a wide range of gamma-ray experiments, the detection system is calibrated by taking a spectrum of a sample with known activities of known nuclides. Then, a spectrum of one or more unknown samples is taken and the energy and activities of its peaks computed from the spectrum and the calibration coefficients. For this experiment, both energy and efficiency calibration functions must be available. The two point linear energy calibration used in the demonstration system is not really adequate and should be replaced by a multipoint polynomial curve fitting routines for energy and efficiency. An additional calibration function which could be added is automatic nuclide identification. A library of nuclides could be kept on the disk where each nuclide included would have a list of the energy of its principal gamma rays. The proper data structure would allow

- searching the library for a particular energy,
- compiling a list of possible nuclides, and
- finding other peaks which should also be present.

Display Function. During all of the other functions, a display of the current spectrum should be available. The user should have control of the display to show a particular section of the specrum on log or linear scale, but the system should have automatic features as well. The system should adjust vertical gain so that the largest feature just reaches the top of the display. During other functions, the display could dynamically indicate the progress of the algorithm. For example, during peak processing; highlight the peak being processed, show the left and right limits used, and display the computed background being subtracted. With a continuous cursor control, light pen, or even cursor control keys, the user should be able to designate a particular peak or a particular portion of the displayed spectrum. A well designed display can make the entire analysis program much easier to use.

Report Function. The final product of any analysis is a written report and the user should not be forced to copy results from a screen into a notebook and later to compile a table of results. A default report should be generated on the hard copy printer by means of a simple command. The report should contain all information derived from the spectrum. For each peak the calibrated energy, intensity, and error limits should be listed as a minimum. The format of the report should be alterable by the user; for simple changes, by keyboard command. For more complex changes, the report writing program can be changed.

Acquire Function. The final function discussed, but usually the first function executed, is the acquisition of data by the multichannel analyzer. This function could be performed by manipulating the front panel controls of the analyzer itself. Alternatively, the Nuclear Data analyzer can be controlled by the microcomputer through the interface. This option is particularly attractive when a series of spectra are to be taken. Then a supervisory program could be written which

- causes the analyzer to acquire a spectrum,
- transfers the spectrum from the multichannel analyzer,
- analyzes the data,
- and preserves the results, both on hard copy and disk.

If the interface were suitably configured, then the acquisition program could also monitor and control other experimental parameters (such as sample movement by pneumatic rabbit).

Summary. Many functions are possible with the equipment proposed in the previous section. All of them require software which will take time to design, write, and implement. The ideas in this section are merely sketches of possible functions which would be useful and attainable. The implementation details need to be determined after a complete study of the features required and desired of each function.

Cost/Benefit Comparison

A decision to buy the equipment and write software for the proposed system should be based not only on what the system will do but also on how much the system will cost. As an alternative approach to meeting system requirements, commercial multichannel analyzers were examined to see if they would have the required features at a comparable cost.

Cost of Proposed System. First, the proposed system was priced, not to obtain an exact cost but rather an upper bound on cost. More accurate costing would depend on the exact system chosen. The items chosen, manufacturer's part number, and catalog price are presented in Table VI.

Commercial Multichannel Analyzers. With the proposed system price available, some comparison shopping was done. Salesmen for three multichannel analyzer manufacturers were contacted by telephone, and the price obtained for an analyzer system with capabilities roughly equivalent to the proposed system.

Canberra Industries, Inc. of Meriden, Connecticut manufactures the Model 80 multichannel analyzer. The Model 80 would be about $13,000 and does perform most of the proposed system functions with cassette tape data storage. However, to provide any user programming, a model designated the PDT-Jupiter would be required. This model contains a Model 80 analyzer and a LSI-11 microcomputer with a disc based operating system. Complete with applications software the system would cost $27,000.

Tracor Northern of Middleton, Wisconsin manufactures the Model TN-1710 Modular Multichannel Analysis System. A Model TN-1710 system is composed of several interconnected modules. Modules were selected for a typical system to fill the requirements stated in the introduction, and the system cost with those modules would be $23,215 including system software.

Nuclear Data Inc. of Schaumberg, Illinois manufactures the Model 66 multichannel analyzer which gathers data only.

Table VI

Catalog Prices of Proposed System Hardware

Item	Manufacturer	Model	Price
Microcomputer	Intel	SBC-80/20-4	$ 925
Disc Controller	Intel	SBX-218	540
High Speed Math	Intel	SBX-331	450
Memory	Intel	SBC-064	2310
Card Cage	Intel	SBC-604	235
Card Cage Extension	Intel	SBC-614	235
Power Supply	Intel	SBC-640	825
Printer	Centronics	730	775
Terminal	Intertec	II	784
Dual Disc Drive	Persci	277	1575
		Total	$8654

(Ref 16,17)

To perform analysis, a Model 680 computer is added. The computer is LSI-11 based with two 8 inch floppy discs. The system cost would be about $35,000. From the salesman's description, this software is more sophisticated than either of the other manufacturer's analyzers or the proposed system. As an example, up to 10 overlapping peaks can be resolved and unfolded. The hardware is also sophisticated with up to 4 analog to digital converters gathering data at the same time spectrum analysis and display are operating.

Comparison. To obtain all of the required system features in a commercial system would cost in excess of $23,000 as compared to proposed system hardware costs of $9,000. The difference is in the software and in the additional capabilities of the commercial equipment. Also, the data gathering function of the multichannel analyzer is a sunk cost for the proposed system. The disadvantage of the proposed system is that it will take time to develop the software. An advantage is the tutorial benefits to be gained by the programmers who write that software. If the system were designed in a proper, structured manner, many students could each add an analyzer function to the system.

IV. Conclusions and Recommendations

Discussion

The primary purpose of this research was to determine whether a microcomputer interfaced with the Nuclear Data multichannel analyzer could perform the analysis of gamma-ray spectra better than the present system. Feasibility was demonstrated by building an interface and writing software to exercise the interface and perform analysis functions. The analysis functions demonstrated were: peak detection, peak location, peak area, and energy calibration. Each of these functions was limited because the microcomputer available for this study could only store 4096 bytes of program. The limitations imposed were primarily assumptions about peak width and the use of a single channel local minimum to define peak boundaries. The software written did perform these limited functions in less than five minutes for one peak and six minutes for fifty peaks.

A secondary purpose was to determine the necessary and desirable characteristics of a microcomputer system to be interfaced with the multichannel analyzer. Hardware characteristics of available components were evaluated against proposed system requirements. Major decisions such as 8 bit versus 16 bit processors and 8 inch floppy discs versus 5 inch floppy discs were outlined and recommendations were made. Software for the proposed system was also discussed. Functions of the analysis system, which could be implemented in software, were explained. The importance of structured design and selection of a language which supports structured programming were emphasized.

The following sections summarize the conclusions of the feasibility demonstration, and the recommendations for the proposed system.

Conclusions

- The Nuclear Data Model 2200 multichannel analyzer has been interfaced to a microcomputer.

- Data can be transferred from the multichannel analyzer to the microcomputer in less than 1 minute for 4096 channels. Nearly all of the time is consumed converting the data to binary form.

- Data transfer followed by peak detection can be accomplished on a 4096-channel, 50-peak spectrum in less than 6 minutes with 32-bit integer arithmetic.

- Program storage of 4096 bytes is not sufficient for any data processing, beyond rudimentary peak location, peak integration, and energy calibration.

- At least 32768 bytes of random access memory are required to analyze 4096 channel spectra.

- A suitable microcomputer with peripherals can be purchased for less than half the cost of the least expensive commercial multichannel analyzer with the required capabilities.

Recommendations

Design a microcomputer system using off-the-shelf assemblies.

- Use a microcomputer based on the Intel 8080 microprocessor to take advantage of software written for this study.
- Obtain 65536 bytes of random access memory.
- Include two 8 inch floppy disk drives.
- Add a high speed math unit.
- Use an existing cathode ray tube terminal with graphics capability; a light pen or joystick is desireable.

Design, build, test, and install an interface between the microcomputer selected and the multichannel analyzer.

Design, write, and test software for the system.

- Use structured, modular programming techniques.
- Use the FORTRAN-77 high level language wherever possible.
- Analyzer functions should be called up by either a command from the terminal or a supervisory program call.
- Devise a program structure which makes additions or changes of modules easy and direct.

Bibliography

1. Series 2200 and 2201 System Analyzers, Instruction Manual. Nuclear Data, Inc. Palatine IL. May 1969

2. SBC 80/20 Hardware Reference Manual. Intel Corp. Santa Clara, CA. 1976. (Order Number 9800317A)

3. Hill, Robert E. Aircrew Modularized Inflight Data Acquisition System. MS Thesis. Air Force Institute of Technology, Wright-Patterson AFB, OH. December 1978. (AFIT/GE/EE/78-28)

4. ISIS II User's Guide. Intel Corp. Santa Clara, CA. 1976.(Order Number 9800306-05)

5. 8080/8085 Assembly Language Programming Manual. Intel Corp. Santa Clara, CA. 1977. (Order Number 9800301C)

6. FORTRAN-80 Programming Manual. Intel Corp. Santa Clara, CA. 1978. (Order Number 980048lA)

7. Clarke, DeFrance. Revisions to the Computer Code GAMMA. Term laboratory report for NE 6.12, Nuclear Engineering Laboratory. School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH. September 1980.

8. Guice, David C. et al. A Study of the Radionuclide Content of the Miami Valley Aqueous Environment. Term research report for NE 6.12, Nuclear Engineering Laboratory. School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH. Spring 1975.

9. VonMeerwall, E. and M.D. Gawlik "Automatic Peak Analysis on Minicomputers." Computer Physics Communications, 5:309-313 19 January 1973.

10. Bevington, Philip R. Data Reduction and Error Analysis for the Physical Sciences. McGraw Hill. New York. 1969.

11. Findley, Robert. Scelbi "8080" Software Gourmet Guide and Cookbook. Scelbi Computer Consulting, Inc. Milford, CT. 1976.

12. Flores, Ivan. The Logic of Computer Arithmetic. Prentice-Hall. Englewood Cliffs, NJ. 1963.

13. USA Standard FORTRAN. American National Standards Institute, Inc. New York. (ANSI X3.9-1966)

14. American National Standard Programming Language FORTRAN. American National Standards Institute, Inc. New York. (ANSI X3.9-1975)

15. Knoll, Glenn F. Radiation Detection and Measurement. John Wiley & Sons. New York. 1979.

16. OEM Price List. Intel Corp. Santa Clara, CA. 14 July 1980 (Updated 1 November 1980)

17. 1980 Catalog. Advanced Computer Products, Inc. PO Box 17329, Irvine, CA.

Appendix A

Program Listings

On the following pages is the compilation or assembly listing of each program module which is part of the demonstration system. In addition, the Fortran equivalent of the PKPROC module is included. The listings are arranged in alphabetical order by module name.

ASM :F1:ASSEM.SRC PAGELENGTH(60) PAGEWIDTH(79)


```
 LOC  OBJ         SEQ        SOURCE STATEMENT

                        $MACROFILE
                        $TITLE('ASSEM - ASSEMBLY LANGUAGE SUBROUTINES FOR GAMMA
                        ')
                        ;
                        ; 17 NOV 80
                        ;
                                NAME    ASSEM
                        ;
                                CSEG
                        ;
                        ;*******************************************
                        ;
                                PUBLIC  OUTCHR
                        OUTCHR:         ;OUTPUT FORTRAN CHARACTER
                                        ;STRING TO CONSOLE
                        ; FORTRAN CALL: CALL OUTCHR('STRING')
                        ; INPUTS:       B,C = ADDRESS OF STRING
                        ;               D,E = LENGTH OF STRING
                        ; CALLS:        CO
                                EXTRN   CO
                        ; DESTROYS:     A,C,D,E,H,L
                        ;
0000 60                         MOV     H,B
0001 69                         MOV     L,C     ; H,L = STRING ADDR
                        TSTDE:          ; TEST REMAINING LENGTH
0002 7A                         MOV     A,D
0003 B3                         ORA     E       ; D,E = 0 ?

0004 C8                         RZ              ;   YES, DONE
0005 4E                         MOV     C,M
0006 CD0000   E                 CALL    CO
0009 23                         INX     H
000A 1B                         DCX     D
000B C30200   C                 JMP     TSTDE
                        ;
                        ;**********************************
                        ;
                                PUBLIC  ININT
                        ININT:          ; INPUT POSITIVE INTEGER FROM CONSOLE
                        ; FORTRAN CALL: CALL ININT(I,ID)
                        ; INPUTS:       B,C = ADDRESS OF I
                        ;               (D,E) = DEFAULT VALUE
                        ; OUTPUTS:      (B,C) <-- 16-BIT VALUE INPUT
                        ; CALLS:        CI,CO,MUL10,OUTSTR,CRLF
                                EXTRN   CI
                        ; DESTROYS:     A,B,C,D,E,H,L
```

```
LOC  OBJ          SEQ         SOURCE STATEMENT

                   46 ;
000E C5            47          PUSH   B         ; SAVE ADDRESS
000F 1A            48          LDAX   D         ; MOVE DEFAULT
0010 02            49          STAX   B         ;  ... TO I
0011 13            50          INX    D
0012 03            51          INX    B
0013 1A            52          LDAX   D
0014 02            53          STAX   B
0015 210000        54          LXI    H,0
0018 0600          55          MVI    B,0
                   56 GET1C:
001A CD0000   E    57          CALL   CI        ; A = CHAR FROM CONSOLE
001D E67F          58          ANI    7FH       ; WITHOUT PARITY (IF ANY)
001F FE0D          59          CPI    CR
0021 CA4600   C    60          JZ     DONE
0024 4F            61          MOV    C,A
0025 CD0000   E    62          CALL   CO        ; ECHO CHAR TO CONSOLE
0028 79            63          MOV    A,C
0029 D630          64          SUI    '0'       ; BETWEEN '0'-'9'
002B DA5200   C    65          JC     INVAL     ;   NO, INVALID
002E FE0A          66          CPI    10
0030 D25200   C    67          JNC    INVAL
0033 CD7300   C    68          CALL   MUL10     ; H,L = H,L * 10
0036 DA5200   C    69          JC     TOOBIG    ; CARRY = OVERFLOW
0039 85            70          ADD    L
003A 6F            71          MOV    L,A
003B 7C            72          MOV    A,H       ; H,L = H,L + DIGIT
003C CE00          73          ACI    0
003E DA5200   C    74          JC     TOOBIG
0041 67            75          MOV    H,A
0042 04            76          INR    B         ; B = COUNT OF CHARS
0043 C31A00   C    77          JMP    GET1C
                   78 DONE:
0046 CD7D00   C    79          CALL   CRLF
0049 EB            80          XCHG             ; D,E = RESULT
004A E1            81          POP    H
004B AF            82          XRA    A
004C B8            83          CMP    B         ; # CHARS = 0 ?
004D C8            84          RZ               ;   YES, DON'T CHANGE IT
004E 73            85          MOV    M,E
004F 23            86          INX    H
0050 72            87          MOV    M,D       ; CALLING ARG <-- RESULT
0051 C9            88          RET
                   89 ;
                   90 INVAL:                    ; CHAR NOT 0-9 OR CR
                   91 TOOBIG:                   ; NUMBER > 2**16
0052 CD7D00   C    92          CALL   CRLF
0055 215F00   C    93          LXI    H,INVMSG
0058 CD9100   C    94          CALL   OUTSTR    ; TELL USER 'INVALID'
```

```
LOC  OBJ          SEQ         SOURCE STATEMENT

005B C1            95          POP     B
005C C30E00   C    96          JMP     ININT   ; TRY AGAIN
005F 494E5641      97 INVMSG:  DB      'INVALID, TRY AGAIN>',0
0063 4C49442C
0067 20545259
006B 20414741
006F 494E3E
0072 00
                   98 ;
                   99 ;***********************************
                  100 ;
                  101 MUL10:           ; MULTIPLY BY 10
                  102 ; INPUTS:        H,L = POSITIVE INTEGER
                  103 ; OUTPUTS:       H,L = H,L * 10
                  104 ; DESTROYS:      D,E
                  105 ;
0073 29           106          DAD     H        ; * 2
0074 D8           107          RC
0075 54           108          MOV     D,H
0076 5D           109          MOV     E,L
0077 29           110          DAD     H        ; * 4
0078 D8           111          RC
0079 29           112          DAD     H        ; * 8
007A D8           113          RC
007B 19           114          DAD     D        ; *8 + *2
007C C9           115          RET
                  116 ;
                  117 ;***********************************
                  118 ;
                  119          PUBLIC  CRLF
                  120 CRLF:            ; CONSOLE <-- CR,IDLE,LF,IDLE
                  121 ; FORTRAN CALL:  CALL CRLF
                  122 ; CALLS:         CO
                  123 ; DESTROYS:      A,C
                  124 ; CALLS:         CO
000D              125 CR       EQU     13       ; CARRIAGE RETURN
000A              126 LF       EQU     10       ; LINE FEED
0016              127 SYN      EQU     22       ; SYNCHRONOUS IDLE
                  128                           ; USED BECAUSE CDC TERMINAL
                  129                           ; EATS CHAR FOLLOWING CR --
                  130                  .        ; SOMETIMES
007D 0E0D         131          MVI     C,CR
007F CD0000   E   132          CALL    CO
0082 0E16         133          MVI     C,SYN
0084 CD0000   E   134          CALL    CO
0087 0E0A         135          MVI     C,LF
0089 CD0000   E   136          CALL    CO
008C 0E16         137          MVI     C,SYN
008E C30000   E   138          JMP     CO       ; LET CO RETURN TO CALLER
```

```
 LOC  OBJ          SEQ          SOURCE STATEMENT

                    139 ;
                    140 ;***************************************
                    141 ;
                    142         PUBLIC  OUTSTR
                    143 OUTSTR:         ; PRINT A CHAR STRING
                    144                 ; ASSEMBLY LANGUAGE CALLED
                    145 ; INPUTS:       H,L = ADDRESS OF STRING
                    146 ;                     TERMINATED BY 0 BYTE
                    147 ; CALLS:        CO
                    148 ; DESTROYS:     A,C,H,L
                    149 ;
 0091 AF            150         XRA     A
 0092 BE            151         CMP     M       ; NEXT CHAR = 0 ?
 0093 C8            152         RZ              ;   YES, DONE
 0094 4E            153         MOV     C,M
 0095 CD0000    E   154         CALL    CO      ;   NO, OUTPUT IT
 0098 23            155         INX     H
 0099 C39100    C   156         JMP     OUTSTR  ; NEXT CHAR
                    157 ;
                    158 ;***************************************
                    159 ;
                    160         PUBLIC  OUTINT
                    161 OUTINT:         ; OUTPUT POSITIVE INTEGER TO CONSOLE
                    162 ; FORTRAN CALL: CALL OUTINT(I,NC)
                    163 ; INPUTS:       B,C = ADDRESS OF I
                    164 ;                     NUMBER TO OUTPUT
                    165 ;               D,E = ADDRESS OF NC
                    166 ;                     MINIMUM # OF CHARS
                    167 ;                     (PAD WITH LEADING BLANKS)
                    168 ; CALLS:        DIVHD,PAD,OUTLP
                    169 ; DESTROYS:     ALL
                    170 ;
 009C 1A            171         LDAX    D
 009D 60            172         MOV     H,B
 009E 69            173         MOV     L,C
 009F 5E            174         MOV     E,M
 00A0 23            175         INX     H       ; D,E = # TO OUTPUT
 00A1 56            176         MOV     D,M
 00A2 0600          177         MVI     B,0     ; B = # DIGITS
 00A4 4F            178         MOV     C,A     ; C = MIN # CHARS
 00A5 7A            179         MOV     A,D
 00A6 B3            180         ORA     E       ; D,E = 0 ?
 00A7 C2AF00    C   181         JNZ     DVLOOP  ;   NO
 00AA 04            182         INR     B       ;   YES, PRINT 1 ZERO
 00AB D5            183         PUSH    D
 00AC C3BF00    C   184         JMP     CKPAD   ; HOW MANY PAD
                    185 DVLOOP:         ; DIVIDE BY 10 AND PUSH REMAINDER
 00AF 210A00        186         LXI     H,10
 00B2 CD9901    C   187         CALL    DIVHD   ; H,L = D,E / H,L
```

```
LOC  OBJ         SEQ        SOURCE STATEMENT

00B5 EB          188         XCHG              ; D,E = QUOTIENT
00B6 7D          189 MOV     A,L
00B7 F5          190         PUSH    PSW       ; SAVE REMAINDER
00B8 33          191         INX     SP        ; ONLY A REGISTER
00B9 04          192         INR     B         ; COUNT DIGIT
00BA 7A          193         MOV     A,D
00BB B3          194         ORA     E         ; D,E = 0 ?
00BC C2AF00   C  195         JNZ     DVLOOP    ;   NO, GET NEXT DIGIT
                 196 CKPAD:                    ; CHECK HOW MANY PAD NEEDED
00BF 79          197         MOV     A,C
00C0 90          198         SUB     B         ; A = # SPACES
00C1 CDDA00   C  199         CALL    PAD
00C4 CDC800   C  200         CALL    OUTLP
00C7 C9          201         RET
                 202 ;
                 203 ;*******************************************
                 204 ;
                 205 OUTLP:            ; POP AND PRINT DIGITS
                 206 ; INPUTS:         B = # OF DIGITS TO PRINT
                 207 ;                 STACK = DIGITS TO PRINT
                 208 ; CALLS:          CO
                 209 ; DESTROYS:       A,B,C,D,E
                 210 ;
00C8 78          211         MOV     A,B       ; # OF DIGITS TO PRINT
00C9 A7          212         ANA     A         ; ... ANY ?
00CA C8          213         RZ                ;     NO
00CB D1          214         POP     D         ; D,E = RETURN ADDRESS
                 215 OUTLP1:
00CC 3B          216         DCX     SP        ; SET UP SP
00CD F1          217         POP     PSW       ; ... POP ONE BYTE INTO A
00CE C630        218         ADI     '0'
00D0 4F          219         MOV     C,A
00D1 CD0000   E  220         CALL    CO
00D4 05          221         DCR     B
00D5 C2CC00   C  222         JNZ     OUTLP1
00D8 D5          223         PUSH    D
00D9 C9          224         RET
                 225 ;
                 226 ;*******************************************
                 227 ;
                 228 PAD:              ; PRINT SPACE FOR EACH COUNT
                 229 ; INPUTS:         A = # OF BLANKS TO PAD WITH
                 230 ;                           0 < A < 128
                 231 ; CALLS:          CO
                 232 ; DESTROYS:       A,C,D
                 233 ;
00DA C8          234         RZ
00DB F8          235         RM
00DC 0E20        236         MVI     C,' '
```

```
LOC  OBJ          SEQ        SOURCE STATEMENT

00DE 57           237            MOV     D,A
                  238 PADLP:
00DF CD0000   E   239            CALL    CO
00E2 15           240            DCR     D
00E3 C2DF00   C   241            JNZ     PADLP
00E6 C9           242            RET
                  243 ;
                  244 ;*******************************************
                  245 ;
                  246            PUBLIC  OUTFLT
                  247 OUTFLT:            ; OUTPUT SIGNED 32-BIT INTEGER
                  248                    ; TO CONSOLE
                  249 ; FORTRAN CALL:  CALL OUTFLT(F,NS*100+NC)
                  250 ; INPUTS:        (B,C) = F - NUMBER TO OUTPUT
                  251 ;                (D,E) = NC - MINIMUM # OF CHARS
                  252 ;                         (PAD WITH LEADING BLANKS)
                  253 ;                         NS - # OF DIGITS AFTER .
                  254 ; DESTROYS:      ALL
                  255 ; CALLS:         FQFLD4,FQFNEG,FQFDV4,PAD,OUTLP,CO
                  256 ;                         DIVHD
                  257 ;
                  258            EXTRN   FQFLD4,FQFNEG,FQFDV4
                  259 ;
00E7 EB           260            XCHG            ; (H,L) = NS*100 + NC
00E8 5E           261            MOV     E,M
00E9 23           262            INX     H
00EA 56           263            MOV     D,M     ; D,E = NS*100 + NC
00EB 216400       264            LXI     H,100
00EE CD9901   C   265            CALL    DIVHD   ; E = NC    L = NS
00F1 7D           266            MOV     A,L
00F2 210000   D   267            LXI     H,NS    ; (H,L) = NS
00F5 77           268            MOV     M,A     ; NS = # DIGITS AFTER  .
00F6 A7           269            ANA     A       ; NS = 0 ?
00F7 CAFB00   C   270            JZ      $+4
00FA 1D           271            DCR     E       ;   NO, NC=NC-1
00FB 23           272            INX     H       ; (H,L) = NC
00FC 73           273            MOV     M,E     ; NC = # CHARS REQUESTED
00FD 50           274            MOV     D,B
00FE 59           275            MOV     E,C
00FF 010300   D   276            LXI     B,ACC
0102 CD0000   E   277            CALL    FQFLD4  ; ACC = NUMBER TO OUTPUT
0105 3A0600   D   278            LDA     ACC+3   ; MSB OF ACC
0108 320F00   D   279            STA     SIGN    ; SAVE FOR LATER
010B A7           280            ANA     A       ; ACC < 0 ?
010C F21301   C   281            JP      OFINIT  ;   NO
010F CD0000   E   282            CALL    FQFNEG
0112 35           283            DCR     M       ; "-" IS ONE CHAR
                  284 OFINIT:
0113 23           285            INX     H       ; (H,L) = ND
```

```
LOC  OBJ          SEQ        SOURCE STATEMENT

0114 3600          286          MVI     M,0      ; ND = # DIGITS ON STACK
                   287 OFLOOP:          ; DIVIDE BY 10 LOOP
0116 CD8801   C    288          CALL    FCOMP0   ; ACC = 0 ?
0119 CA2B01   C    289          JZ      OFCHK    ;   YES
011C 119501   C    290          LXI     D,TEN
011F CD0000   E    291          CALL    FQFDV4   ; ACC = ACC/10
0122 3A0700   D    292          LDA     EXT      ; EXT = REMAINDER
0125 F5            293          PUSH    PSW      ; SAVE REMAINDER ON
0126 33            294          INX     SP       ; .. ONE BYTE OF STACK
0127 34            295          INR     M        ; BUMP DIGIT COUNT
0128 C31601   C    296          JMP     OFLOOP
                   297 OFCHK:
012B 3A0000   D    298          LDA     NS
012E 96            299          SUB     M        ; A = NS - ND
012F F25101   C    300          JP      OFZERO   ; NOT ENOUGH DIGITS
0132 3A0100   D    301          LDA     NC
0135 96            302          SUB     M        ; A = NC - ND
0136 CDDA00   C    303          CALL    PAD      ; PAD W/ NC-ND BLANKS
0139 CD7D01   C    304          CALL    OFSIGN
013C 7E            305          MOV     A,M
013D 2B            306          DCX     H
013E 2B            307          DCX     H        ; (H,L) = NS
013F 96            308          SUB     M
0140 47            309          MOV     B,A      ; B = ND - NS
0141 CDC800   C    310          CALL    OUTLP    ; PRINT DICITS BEFORE .
0144 7E            311          MOV     A,M
0145 A7            312          ANA     A        ; DIGITS AFTER . ?
0146 C8            313          RZ               ;   NO
0147 0E2E          314          MVI     C,'.'
0149 CD0000   E    315          CALL    CO
014C 46            316          MOV     B,M
014D CDC800   C    317          CALL    OUTLP    ; PRINT DIGITS AFTER DOT
0150 C9            318          RET
                   319 OFZERO:          ; FEWER DIGITS THAN DECIMAL
                   320                  ; PLACES.  SUPPLY LEADING ZEROES
0151 47            321          MOV     B,A      ; B = DEFICIT
0152 2B            322          DCX     H        ; (H,L) = NC
0153 7E            323          MOV     A,M
0154 2B            324          DCX     H        ; (H,L) = NS
0155 96            325          SUB     M
0156 3D            326          DCR     A        ; A = NC-NS-1
0157 CDDA00   C    327          CALL    PAD      ; PAD WITH BLANKS
015A CD7D01   C    328          CALL    OFSIGN
015D 0E30          329          MVI     C,'0'    ; ZERO BEFORE DOT
015F CD0000   E    330          CALL    CO
0162 7E            331          MOV     A,M
0163 A7            332          ANA     A        ; DIGITS AFTER DOT ?
0164 C8            333          RZ               ;   NO
0165 0E2E          334          MVI     C,'.'
```

```
 LOC  OBJ          SEQ          SOURCE STATEMENT

0167 CD0000   E    335          CALL    CO
016A 23            336          INX     H
016B 23            337          INX     H
016C 0E30          338          MVI     C,'0'
                   339 OFOLP:
016E 05            340          DCR     B        ; MORE ZEROES NEEDED ?
016F FA7801   C    341          JM      OFDONE   ;   NO, PRINT DIGITS
0172 CD0000   E    342          CALL    CO       ;   YES, PRINT '0'
0175 C36E01   C    343          JMP     OFOLP
                   344 OFDONE:
0178 46            345          MOV     B,M
0179 CDC800   C    346          CALL    OUTLP
017C C9            347          RET
                   348 ;
                   349 OFSIGN:              ; PRINT "-" IF ARGUMENT NEGATIVE
017D 3A0F00   D    350          LDA     SIGN
0180 A7            351          ANA     A        ; ORIGINAL ACC < 0 ?
0181 F0            352          RP               ;   NO
0182 0E2D          353          MVI     C,'-'
0184 CD0000   E    354          CALL    CO       ;   YES
0187 C9            355          RET
                   356 ;
                   357 FCOMP0:              ; COMPARE ACC WITH ZERO
0188 E5            358          PUSH    H
0189 210300   D    359          LXI     H,ACC
018C 7E            360          MOV     A,M
                   361          REPT    3
                   362          INX     H
                   363          ORA     M
                   364          ENDM
018D 23            365+         INX     H
018E B6            366+         ORA     M
018F 23            367+         INX     H
0190 B6            368+         ORA     M
0191 23            369+         INX     H
0192 B6            370+         ORA     M
0193 E1            371          POP     H
0194 C9            372          RET              ; Z STATUS TELLS TALE
                   373 ;
0195 0A00          374 TEN:     DW      10,0
0197 0000
                   375          DSEG
                   376 ; THE ORDER OF THE NEXT 3 ITEMS IS IMPORTANT
0001               377 NS:      DS      1
0001               378 NC:      DS      1
0001               379 ND:      DS      1
                   380 ; THE ORDER OF THE NEXT TWO ITEMS IS IMPORTANT
0004               381 ACC:     DS      4
0008               382 EXT:     DS      8
```

```
 LOC  OBJ          SEQ        SOURCE STATEMENT

 0001               383 SIGN:   DS      1
                    384         CSEG
                    385 ;
                    386 ;*******************************************
                    387 ;
                    388 DIVHD:              ; DIVIDE D,E BY H,L
                    389 ; INPUTS:          D,E = DIVIDEND
                    390 ;                  H,L = DIVISOR
                    391 ; OUTPUTS:         D,E = REMAINDER
                    392 ;                  H,L = QUOTIENT
                    393 ; DESTROYS:        A
                    394 ;
 0199 C5            395         PUSH    B
 019A EB            396         XCHG
 019B 42            397         MOV     B,D
 019C 4B            398         MOV     C,E       ; B,C = DIVISOR
 019D 110000        399         LXI     D,0       ; D,E,H,L = DIVIDEND:QUOTIENT
 01A0 3E10          400         MVI     A,16      ; A = LOOP COUNT
                    401 D1:
 01A2 F5            402         PUSH    PSW
 01A3 29            403         DAD     H         ; LEFT SHIFT
 01A4 7B            404         MOV     A,E       ; ... D,E,H,L
 01A5 17            405         RAL               ; ... ONE BIT
 01A6 5F            406         MOV     E,A
 01A7 7A            407         MOV     A,D
 01A8 17            408         RAL
 01A9 57            409         MOV     D,A
 01AA 7B            410         MOV     A,E       ; D,E - B,C
 01AB 91            411         SUB     C
 01AC 5F            412         MOV     E,A
 01AD 7A            413         MOV     A,D
 01AE 98            414         SBB     B
 01AF 57            415         MOV     D,A       ; ABLE TO SUBTRACT ?
 01B0 D2B701     C  416         JNC     D2        ;   YES
 01B3 EB            417         XCHG              ;   NO, ADD IT BACK
 01B4 09            418         DAD     B
 01B5 EB            419         XCHG
 01B6 23            420         INX     H         ; KEEP INVERTED QUOTIENT
                    421 D2:
 01B7 F1            422         POP     PSW
 01B8 3D            423         DCR     A         ; LOOP COUNT -1
 01B9 C2A201     C  424         JNZ     D1        ;   DO ANOTHER BIT
 01BC C1            425         POP     B
 01BD 7D            426         MOV     A,L       ; COMPLEMENT
 01BE 2F            427         CMA               ; ... QUOTIENT
 01BF 6F            428         MOV     L,A       ; ... IN H,L
 01C0 7C            429         MOV     A,H
 01C1 2F            430         CMA
 01C2 67            431         MOV     H,A
```

```
  LOC  OBJ         SEQ        SOURCE STATEMENT

  01C3 C9           432        RET
                    433 ;
                    434        END

PUBLIC SYMBOLS
CRLF   C 007D   ININT  C 000E   OUTCHR C 0000   OUTFLT C 00E7
OUTINT C 009C   OUTSTR C 0091

EXTERNAL SYMBOLS
CI     E 0000   CO     E 0000   FQFDV4 E 0000   FQFLD4 E 0000
FQFNEG E 0000

USER SYMBOLS
ACC    D 0003   CI     E 0000   CKPAD  C 00BF   CO     E 0000
CR     A 000D   CRLF   C 007D   D1     C 01A2   D2     C 01B7
DIVHD  C 0199   DONE   C 0046   DVLOOP C 00AF   EXT    D 0007
FCOMP0 C 0188   FQFDV4 E 0000   FQFLD4 E 0000   FQFNEG E 0000
GET1C  C 001A   ININT  C 000E   INVAL  C 0052   INVMSG C 005F
LF     A 000A   MUL10  C 0073   NC     D 0001   ND     D 0002
NS     D 0000   OFOLP  C 016E   OFCHK  C 012B   OFDONE C 0178
OFINIT C 0113   OFLOOP C 0116   OFSIGN C 017D   OFZERO C 0151
OUTCHR C 0000   OUTFLT C 00E7   OUTINT C 009C   OUTLP  C 00C8
OUTLP1 C 00CC   OUTSTR C 0091   PAD    C 00DA   PADLP  C 00DF
SIGN   D 000F   SYN    A 0016   TEN    C 0195   TOOBIG C 0052
TSTDE  C 0002

ASSEMBLY COMPLETE,  NO ERRORS
```

FORTRAN COMPILER   ENCAL - ENERGY CALIBRATION   20 NOV 80 PAGE 1

ISIS-II FORTRAN-80 V1.0 COMPILATION OF PROGRAM UNIT ENCAL
OBJECT MODULE PLACED IN ENCAL.OBJ
COMPILER INVOKED BY: :F1:FORT80 ENCAL.FOR PAGELENGTH(54)

```
$PAGEWIDTH(73) DATE(20 NOV 80)
$TITLE('ENCAL - ENERGY CALIBRATION')
        SUBROUTINE ENCAL
C
C ASK USER FOR PEAK NUMBER AND ENERGY OF TWO PEAKS
C COMPUTE KEV/CHANNEL BASED ON DIFERENCE
C COMPUTE OFFSET BASED ON FIRST PEAK
C PRINT ENERGY FOR ALL PEAKS
C
C SCALING:
C   REAL VARIABLES ARE STORED AS 32-BIT INTEGERS AND ARE
C   SCALED BY POWERS OF TEN TO ALLOW FOR NON-INTEGER VALUES
C
C VARIABLE      SCALING         USE
C --------      -------         ---
C   CHANL         10            CHANNEL NUMBER
C   PKERGY        10            ENERGY IN KEV
C   COEFF(1)      10            OFFSET IN KEV
C   COEFF(2)      10**4         KEV/CHANNEL
C
        COMMON  NCHANL,NPEAKS,COEFF(8)
        COMMON  CHANL(50),TOTAL(50),AREA(50),PKERGY(50),PGPS(50)
        COMMON  TIME,COUNTS(4095)
C
        IF(NPEAKS.LT.3) RETURN
        CALL CRLF
        CALL OUTCHR('ENERGY CALIBRATION')
        CALL CRLF
10      CALL OUTCHR('FIRST PEAK? ')
        CALL ININT(NPK1)
        IF(NPK1.LT.1.OR.NPK1.GT.NPEAKS) GO TO 10
        CALL OUTCHR('ENERGY? ')
        CALL ININT(IENRGY)
        PKERGY(NPK1)=IENRGY*10
20      CALL OUTCHR('SECOND PEAK? ')
        CALL ININT(NPK2)
        IF(NPK2.LT.1.OR.NPK2.GT.NPEAKS.OR.NPK1.EQ.NPK2) GO TO 20
        CALL OUTCHR('ENERGY? ')
        CALL ININT(IENRGY)
        PKERGY(NPK2)=IENRGY*10
C COMPUTE KEV/CHANNEL AND OFFSET
        SCALE=10000
        COEFF(2)=SCALE*(PKERGY(NPK2)-PKERGY(NPK1))
        COEFF(2)=COEFF(2)/(CHANL(NPK2)-CHANL(NPK1))
        COEFF(1)=PKERGY(NPK1)-CHANL(NPK1)*COEFF(2)/SCALE
```

```
          CALL OUTCHR('ENERGY  =  ')
          CALL OUTFLT(COEFF(1),100)
          CALL OUTCHR('  +  ')
          CALL OUTFLT(COEFF(2),600)
          CALL OUTCHR(' * CHANNEL')
          CALL CRLF
C PRINT ENERGY OF EACH PEAK
          CALL CRLF
          CALL OUTCHR('PEAK CHANNEL ENERGY')
          CALL CRLF
          DO 100 I=1,NPEAKS
            CALL OUTINT(I,3)
            CALL OUTFLT(CHANL(I),108)
            PKERGY(I)=COEFF(1)+COEFF(2)*CHANL(I)/SCALE
            CALL OUTFLT(PKERGY(I),108)
            CALL CRLF
  100       CONTINUE
          CALL CRLF
          RETURN
          END
```

MODULE INFORMATION:

```
    CODE AREA SIZE     = 02A2H    674D
    VARIABLE AREA SIZE = 0022H     34D
    MAXIMUM STACK SIZE = 0004H      4D
    65 LINES READ

    0 PROGRAM ERROR(S) IN PROGRAM UNIT ENCAL

    0 TOTAL PROGRAM ERROR(S)
    END OF FORTRAN COMPILATION
```

```
    -1:GA'MA.SRC PAGELENGTH(60) PAGEWIDTH(79)

ISIS-II 8080/8085 MACRO ASSEMBLER, V2.0          GAMMA    PAGE    1
GAMMA - MAIN ROUTINE OF MICROCOMPUTER GAMMA ANALYSIS PROGRAM

 LOC  OBJ         SEQ          SOURCE STATEMENT

                    1 $MACROFILE
                    2 $TITLE('GAMMA - MAIN ROUTINE OF MICROCOMPUTER GAMMA ANA
                      LYSIS PROGRAM')
                    3 ;
                    4 ; 13 JAN 81
                    5 ;
                    6 ; CALL ROUTINES IN SEQUENCE
                    7 ;
                    8          NAME    GAMMA
                    9          PUBLIC  GAMMA
                   10 ;
                   11 ;*******************************************
                   12 ;
                   13          CSEG
                   14 GAMMA:
                   15          EXTRN   GETDAT,PKSRCH,ENCAL
 0000 CD0000   E   16          CALL    GETDAT
 0003 CD0000   E   17          CALL    PKSRCH
 0006 CD0000   E   18          CALL    ENCAL
 0009 C30000   C   19          JMP     GAMMA
                   20 ;
                   21          END

PUBLIC SYMBOLS
GAMMA  C 0000

EXTERNAL SYMBOLS
ENCAL  E 0000    GETDAT E 0000    PKSRCH E 0000

USER SYMBOLS
ENCAL  E 0000    GAMMA  C 0000    GETDAT E 0000    PKSRCH E 0000

ASSEMBLY COMPLETE,  NO ERRORS
```

```
ASM :F1:GETDAT.SRC PAGELENGTH(60) PAGEWIDTH(79)

ISIS-II 8080/8085 MACRO ASSEMBLER, V2.0        GETDAT    PAGE    1
GETDAT - GET DATA FROM MULTI-CHANNEL ANALYZER

 LOC  OBJ        SEQ        SOURCE STATEMENT

                 $MACROFILE
                 $TITLE('GETDAT - GET DATA FROM MULTI-CHANNEL ANALYZER')
                 ;
                 ; 13 JAN 81
                 ;
                 ; 1. ASK USER "HOW MANY CHANNELS?"
                 ; 2. INPUT CHANNELS FROM MCA
                 ; 3. CONVERT FROM BCD TO STORAGE FORM (FLOATING POINT)
                 ;
                         NAME    GETDAT
                         PUBLIC  GETDAT
                         EXTRN   OUTSTR,ININT
                         CSEG
                 ;
                 ; I/O EQUATES
                 ;
 00D6            LED     EQU     0D6H     ; LED ON SBC BOARD
 00E5            DISPL   EQU     0E5H     ; HEX DIGIT ON INTERFACE
 00E6            OUTBYT  EQU     0E6H     ; OUTPUT ALL 5 BITS + 3 SPARE
 00E7            OUTBIT  EQU     0E7H     ; OUTPUT ONE BIT
 00E8            IN0     EQU     0E8H     ; 10,1 DIGITS
 00E9            IN1     EQU     0E9H     ; 1000,100 DIGITS
 00EA            IN2     EQU     0EAH     ; 100K,10K DIGITS
                 ;
                 ; OUTPUT BITS
                 ;
 0000            MUXSEL  EQU     0        ; SELECT ADDRESS/DATA
 0002            READY2  EQU     2        ; REQUEST NEXT CHANNEL
 0003            EXTA    EQU     3        ; ACQUIRE MODE
 0004            EXTR    EQU     4        ; READOUT MODE
                 ;
 0000            ADDRES  EQU     0
 0001            DATA    EQU     1
 0000            HI      EQU     0        ; BIT INVERTED BY DRIVER
 0001            LO      EQU     1
                 ;
                 BIT     MACRO   BITNO,STATE
                         MVI     A,BITNO SHL 1 OR STATE
                         OUT     OUTBIT
                         ENDM
                 ;
                 ;*****************************************************
                 ;
                 GETDAT:
                 ;
                 ; ASK USER "HOW MANY CHANNELS?"
```

```
LOC  OBJ          SEQ        SOURCE STATEMENT

0000 21B600   C    47         LXI    H,HOWMNY
0003 CD0000   E    48         CALL   OUTSTR  ; ASK QUESTION
0006 11B400   C    49         LXI    D,ZERO  ; DEFAULT, N=0
0009 010000   D    50         LXI    B,N
000C CD0000   E    51         CALL   ININT   ; CALL ININT(N,0)
000F 2A0000   D    52         LHLD   N
0012 7C            53         MOV    A,H
0013 B5            54         ORA    L       ; N = 0 ?
0014 C8            55         RZ             ;   YES, RETURN
0015 2A0000   D    56         LHLD   N
0018 220090        57         SHLD   NCHANL  ;   NO, NCHANL = N
                   58 ;
                   59 ; INITIALIZE FOR INPUT FROM MCA
                   60         BIT    EXTR,LO ; READOUT MODE
001B 3E09          61+        MVI    A,EXTR SHL 1 OR LO
001D D3E7          62+        OUT    OUTBIT
001F 210C94        63         LXI    H,TIME  ; ADDRESS OF FIRST CHAN
0022 220200   D    64         SHLD   ADDR
                   65 ;
                   66 ; LOOP THROUGH CHANNELS
                   67 CHLOOP:
0025 CD5500   C    68         CALL   CMDWAT
0028 CD6500   C    69         CALL   CHIN
                   70         BIT    READY2,HI ; START NEXT ACCESS
002B 3E04          71+        MVI    A,READY2 SHL 1 OR HI
002D D3E7          72+        OUT    OUTBIT
002F CD8B00   C    73         CALL   STORE
0032 2A0000   D    74         LHLD   N
0035 2B            75         DCX    H
0036 220000   D    76         SHLD   N
0039 7C            77         MOV    A,H
003A B5            78         ORA    L       ; N = 0 ?
003B CA4900   C    79         JZ     DONE
003E C32500   C    80         JMP    CHLOOP
                   81         REPT   8       ; PATCH SPACE *DEBUG*
                   82         NOP
                   83         ENDM
0041 00            84+        NOP
0042 00            85+        NOP
0043 00            86+        NOP
0044 00            87+        NOP
0045 00            88+        NOP
0046 00            89+        NOP
0047 00            90+        NOP
0048 00            91+        NOP
                   92 ;
                   93 DONE:          ; RETURN MCA TO USER
                   94         BIT    EXTR,HI ; TURN OFF READ-OUT
0049 3E08          95+        MVI    A,EXTR SHL 1 OR HI
```

```
LOC  OBJ          SEQ        SOURCE STATEMENT

004B D3E7          96+        OUT     OUTBIT
                   97         BIT     READY2,LO ; SIGNAL DONE CHANNEL
004D 3E05          98+        MVI     A,READY2 SHL 1 OR LO
004F D3E7          99+        OUT     OUTBIT
0051 AF           100         XRA     A
0052 D3E6         101         OUT     OUTBYT  ; ALL LINES HIGH
0054 C9           102         RET
                  103 ;
                  104 CMDWAT:             ; WAIT FOR COMMAND LINE TO GO LOW
                  105         BIT     READY2,LO ; DROP READY2
0055 3E05         106+        MVI     A,READY2 SHL 1 OR LO
0057 D3E7         107+        OUT     OUTBIT
                  108         BIT     MUXSEL,ADDRES
0059 3E00         109+        MVI     A,MUXSEL SHL 1 OR ADDRES
005B D3E7         110+        OUT     OUTBIT
005D DBEA         111 WTLOOP: IN      IN2
005F E602         112         ANI     10B     ; COMMAND ?
0061 CA5D00   C   113         JZ      WTLOOP  ;   HIGH, WAIT FOR LOW
0064 C9           114         RET
                  115 ;
                  116 CHIN:               ; INPUT CHANNEL TO DATA
                  117         BIT     MUXSEL,DATA
0065 3E01         118+        MVI     A,MUXSEL SHL 1 OR DATA
0067 D3E7         119+        OUT     OUTBIT
0069 210400   D   120         LXI     H,COUNT ; ADDRESS OF HOLD AREA
006C DBE8         121         IN      IN0
006E CD7C00   C   122         CALL    BCD     ; CONVERT 2 BCD DIGITS
0071 DBE9         123         IN      IN1
0073 CD7C00   C   124         CALL    BCD
0076 DBEA         125         IN      IN2
0078 CD7C00   C   126         CALL    BCD
007B C9           127         RET
                  128 ;
                  129 ;*********************************************
                  130 ;
                  131 BCD:                ; SEPARATE 2 DIGITS INTO 2 BYTES
                  132 ; INPUTS:           A = 2 BCD DIGITS
                  133 ;                   H,L = ADDRESS TO STORE 2 BYTES
                  134 ; OUTPUTS:          H,L = H,L + 2
                  135 ; DESTROYS:         A
                  136 ;
007C 2F           137         CMA             ; DATA WAS INVERTED
007D F5           138         PUSH    PSW
007E CD8600   C   139         CALL    CHXD    ; LEAST SIGNIF DIGIT
0081 F1           140         POP     PSW
                  141         REPT    4
                  142         RRC
                  143         ENDM            ; MOST SIGNIF DIGIT
0082 0F           144+        RRC
```

```
LOC  OBJ           SEQ        SOURCE STATEMENT

0083 0F             145+        RRC
0084 0F             146+        RRC
0085 0F             147+        RRC
0086 E60F           148 CHXD:   ANI     0FH
0088 77             149         MOV     M,A
0089 23             150         INX     H
008A C9             151         RET
                    152 ;
                    153 ;*******************************************
                    154 ;
                    155 STORE:              ; STORE COUNT IN 4 BYTES RAM
                    156 ; INPUTS:           COUNT = 6 DIGITS
                    157 ;                   ADDR = ADDRESS TO STORE RESULT
                    158 ; OUTPUTS:          ADDR = ADDR + 4
                    159 ;                   (ADDR) = CHANNEL DATA
                    160
                    161 ; CALLS:            FQFCLR,FQFAD4,FMUL10
                    162 ;
                    163 ; NOTE:  THE FQF... ROUTINES OPERATE ON A
                    164 ;         12-BYTE LOGICAL REGISTER.
                    165 ;   BYTES 0-3 ARE ACCUMULATOR(ACC)
                    166 ;          4-7     EXTENSION(EXT)
                    167 ;          8-11    AUXILARY(AUX)
                    168 ; THIS LOGICAL REGISTER IS SET WITH THE ACC ON
                    169 ; THE CURRENT CHANNEL COUNT.
                    170 ; EXT & AUX ARE ON FUTURE CHANNELS.
                    171 ;
                    172 ; DESTROYS:         ALL
                    173         EXTRN   FQFCLR,FQFAD4,FMUL10
008B 2A0200    D    174         LHLD    ADDR
008E 44             175         MOV     B,H     ; (B,C) = ACC = COUNTS(I)
008F 4D             176         MOV     C,L
                    177         REPT    4
                    178         INX     H
                    179         ENDM
0090 23             180+        INX     H
0091 23             181+        INX     H
0092 23             182+        INX     H
0093 23             183+        INX     H
0094 220200    D    184         SHLD    ADDR    ; ADDR = ADDR + 4
0097 CD0000    E    185         CALL    FQFCLR  ; ZERO 12 BYTES AT B,C
009A 3E06           186         MVI     A,6     ; A = LOOP COUNT
009C 210800         187         LXI     H,8
009F 09             188         DAD     B
00A0 EB             189         XCHG            ; (D,E) = AUX REG
00A1 210900    D    190         LXI     H,COUNT+5 ; (H,L) = MSD
                    191 STORLP:             ; CONVERT 6 DIGITS TO BINARY
00A4 F5             192         PUSH    PSW
00A5 7E             193         MOV     A,M     ; MOVE DIGIT FROM COUNT
```

```
 LOC  OBJ           SEQ          SOURCE STATEMENT

 00A6 12             194          STAX    D       ;  ... TO AUX REG
 00A7 CD0000    E    195          CALL    FQFAD4  ; ACC = ACC + AUX
 00AA F1             196          POP     PSW
 00AB 3D             197          DCR     A       ; MORE DIGITS ?
 00AC C8             198          RZ              ;   NO
 00AD CD0000    E    199          CALL    FMUL10  ; ACC = ACC * 10
 00B0 2B             200          DCX     H       ; (H,L) = NEXT DIGIT
 00B1 C3A400    C    201          JMP     STORLP  ; DO NEXT DIGIT
                     202 ;
                     203 ;**********************************************
                     204 ;
 00B4 0000           205 ZERO:    DW      0
 00B6 47455420       206 HOWMNY:  DB      'GET HOW MANY CHANNELS? ',0
 00BA 484F5720
 00BE 4D414E59
 00C2 20434841
 00C6 4E4E454C
 00CA 533F20
 00CD 00
                     207          DSEG
 0002                208 N:       DS      2       ; # CHANNELS TO GET
 0002                209 ADDR:    DS      2       ; ADDR OF NEXT CHANNEL
 0006                210 COUNT:   DS      6       ; TEMP HOLD BCD DIGITS
                     211 ;
                     212          ASEG
 9000                213          ORG     9000H   ; LOCATE // TO THIS ADDR
 0002                214 NCHANL:  DS      2
 0002                215 NPEAKS:  DS      2
 0020                216 COEFF:   DS      8*4
 00C8                217 CHANL:   DS      50*4
 00C8                218 TOTAL:   DS      50*4
 00C8                219 AREA:    DS      50*4
 00C8                220 PKERGY:  DS      50*4
 00C8                221 PGPS:    DS      50*4
 0004                222 TIME:    DS      4
 3FFC                223 COUNTS:  DS      4095*4
 0008                224          DS      12-4    ; ROOM TO CLEAR 4096TH
                     225                          ; CHANNEL
                     226 ;
                     227          END

PUBLIC SYMBOLS
GETDAT C 0000

EXTERNAL SYMBOLS
FMUL10 E 0000    FQFAD4 E 0000    FQFCLR E 0000    ININT  E 000?
OUTSTR E 0000

USER SYMBOLS
```

```
ADDR   D 0002   ADDRES A 0000   AREA   A 91B4   BCD    C 007C
BIT    + 0000   CHANL  A 9024   CHIN   C 0065   CHLOOP C 0025
CHXD   C 0086   CMDWAT C 0055   COEFF  A 9004   COUNT  D 0004
COUNTS A 9410   DATA   A 0001   DISPL  A 00E5   DONE   C 0049
EXTA   A 0003   EXTR   A 0004   FMUL10 E 0000   FQFAD4 E 0000
FQFCLR E 0000   GETDAT C 0000   HI     A 0000   HOWMNY C 00B6
IN0    A 00E8   IN1    A 00E9   IN2    A 00EA   ININT  E 0000
LED    A 00D6   LO     A 0001   MUXSEL A 0000   N      D 0000
NCHANL A 9000   NPEAKS A 9002   OUTBIT A 00E7   OUTBYT A 00E6
OUTSTR E 0000   PGPS   A 9344   PKERGY A 927C   READY2 A 0002
STORE  C 008B   STORLP C 00A4   TIME   A 940C   TOTAL  A 90EC
WTLOOP C 005D   ZERO   C 00B4

ASSEMBLY COMPLETE,  NO ERRORS
```

ASM :F1:INT4.SRC PAGELENGTH(60)


```
 LOC  OBJ        SEQ        SOURCE STATEMENT

                      $MACROFILE PAGEWIDTH(79)
                      $TITLE('INT4 - MULTI-BYTE INTEGER ARITHMETIC ROUTINES')
                      ;
                      ; 13 JAN 81
                      ;
                              NAME    INT4
                      ;
                      ; PARAMETER:
                      ;   P IS THE NUMBER OF 8-BIT BYTES USED
                      ;   IN ALL OPERATIONS.
                      ;
 0004                 P       EQU     4        ; 4-BYTE OPERANDS
                      ;
                      ; ARGUMENTS:
                      ;
                      ; REGISTERS B,C CONTAIN THE ADDRESS OF
                      ; A RAM AREA AT LEAST 3*P BYTES LONG WHICH
                      ; IS LOGICALLY DIVIDED INTO THREE "REGISTERS"
                      ;
                      ; 1. THE ACCUMULATOR (ACC) CONTAINS FIRST
                      ;    OPERAND ON ENTRY AND THE RESULT ON EXIT.
                      ;
                      ; 2. THE EXTENSION (EXT) IS USED AS AN EXTENSION
                      ;    OF THE ACC TO CONTAIN TEMPORARY RESULTS.
                      ;
                      ; 3. THE AUXILIARY (AUX) IS USED TO STORE
                      ;    THE SECOND OPERAND IF THAT OPERAND
                      ;    NEEDS TO BE MODIFIED.
                      ;
                      ; REGISTERS D,E CONTAIN THE ADDRESS OF
                      ; THE SECOND OPERAND WHICH IS "P" BYTES
                      ; LONG.
                      ;
                      ; ALL REGISTERS ARE RESTORED ON EXIT
                      ; WITH EXCEPTION OF A-REGISTER FOR COMPARE
                      ; OPERATION.
                      ;
                      ;
                              CSEG
                      ;
                      ;*******************************************
                      ;
                              PUBLIC  FQFAD4
                      FQFAD4:         ; ADD OPERATION
                      ; CALLS:          SAVE
                      ;
```

```
LOC  OBJ           SEQ         SOURCE STATEMENT

0000 CD3C00    C    47          CALL    SAVE
0003 60             48          MOV     H,B
0004 69             49          MOV     L,C      ; (H,L)=ACC
0005 0603           50          MVI     B,P-1    ; B = LOOP COUNT;
0007 A7             51          ANA     A        ; CARRY MUST START CLEAR
                    52 ADDLP:
0008 1A             53          LDAX    D
0009 8E             54          ADC     M        ;(H,L)=(H,L)+(D,E)+CARRY
000A 77             55          MOV     M,A
000B 23             56          INX     H
000C 13             57          INX     D
000D 05             58          DCR     B        ; ANY MORE ?
000E C20800    C    59          JNZ     ADDLP    ;   YES, LOOP
0011 1A             60          LDAX    D        ; MOST SIGNIF BYTE
0012 4E             61          MOV     C,M      ; SAVE OLD SIGN
0013 89             62          ADC     C
0014 77             63          MOV     M,A
0015 1A             64          LDAX    D        ; SIGN OF OP.MSB
0016 A9             65          XRA     C        ; SIGN: OP = OLD ACC ?
0017 F8             66          RM               ;   NO, CAN'T OVERFLOW
0018 7E             67          MOV     A,M      ; SIGN ACC, NEW = OLD ?
0019 A9             68          XRA     C
001A F0             69          RP               ;   YES, OK
001B C31E00    C    70          JMP     FQFERH   ;   NO, OVERFLOW
                    71 ;
                    72 ;*******************************************
                    73 ;
                    74          PUBLIC  FQFERH
                    75 FQFERH:          ; REPORT ERROR TO USER
                    76                  ; AND CALL OPER SYST
                    77 ; CALLS:         OUTSTR,CRLF
                    78          EXTRN   OUTSTR,CRLF
                    79 ;
001E 212800    C    80          LXI     H,ERRMSG
0021 CD0000    E    81          CALL    OUTSTR
0024 CD0000    E    82          CALL    CRLF
0027 CF             83          RST     1        ; INVOKE OPER SYSTEM
                    84 ;
0028 41524954       85 ERRMSG: DB       'ARITHMETIC OVERFLOW',0
002C 484D4554
0030 4943204F
0034 56455246
0038 4C4F57
003B 00
                    86 ;
                    87 ;*************************************
                    88 ;
                    89 SAVE:            ; SAVE ALL REGISTERS
003C E3             90          XTHL             ; SAVE H,L, GET CALLER
```

```
 LOC  OBJ          SEQ        SOURCE STATEMENT

 003D D5            91          PUSH    D
 003E C5            92          PUSH    B
 003F F5            93          PUSH    PSW
 0040 CD4800    C   94          CALL    JUMP      ; PUT RETURN ADDR ON STAK
                    95 ; RETURN FROM CALLING ROUTINE COMES HERE
                    96 ; RESTORE REGISTERS AND RETURN
 0043 F1            97          POP     PSW
 0044 C1            98          POP     B
 0045 D1            99          POP     D
 0046 E1           100          POP     H
 0047 C9           101          RET
 0048 E9           102 JUMP:    PCHL              ; JUMP TO CALLER
                   103 ;
                   104 ;****************************************
                   105 ;
                   106          PUBLIC  FQFSB4
                   107 FQFSB4:          ; SUBTRACT OPERATION
                   108 ; CALLS:         SAVE
 0049 CD3C00    C  109          CALL    SAVE
 004C 60           110          MOV     H,B
 004D 69           111          MOV     L,C       ; (D,E) = ACC
 004E EB           112          XCHG              ; (H,L) = OP
 004F 0603         113          MVI     B,P-1     ; B = LOOP COUNT
 0051 A7           114          ANA     A         ; BORROW STARTS CLEAR
                   115 SUBLP:
 0052 1A           116          LDAX    D
 0053 9E           117          SBB     M         ;(D,E)=(D,E)-(H,L)-BORROW
 0054 12           118          STAX    D
 0055 23           119          INX     H
 0056 13           120          INX     D
 0057 05           121          DCR     B         ; MORE BYTES ?
 0058 C25200    C  122          JNZ     SUBLP     ;   YES
 005B 1A           123          LDAX    D         ; MOST SIGNIF BYTE
 005C 4F           124          MOV     C,A
 005D 9E           125          SBB     M
 005E 12           126          STAX    D
 005F 7E           127          MOV     A,M       ; SIGN OF OP.MSB
 0060 A9           128          XRA     C         ; SIGN: OLD ACC = OP ?
 0061 F0           129          RP                ;   YES, CAN'T OVERFLOW
 0062 1A           130          LDAX    D
 0063 A9           131          XRA     C         ; SIGN ACC, OLD = NEW ?
 0064 F0           132          RP                ;   YES, OK
 0065 C31E00    C  133          JMP     FQFERH    ;   NO, OVERFLOW
                   134 ;
                   135 ;****************************************
                   136 ;
                   137          PUBLIC  FQFNEG
                   138 FQFNEG:          ; TWO'S COMPLEMENT OPERATION
                   139 ; INPUT; CALLS: SAVE
```

```
LOC  OBJ          SEQ         SOURCE STATEMENT

                   140 ;
0068 CD3C00    C   141          CALL    SAVE
006B 60            142          MOV     H,B
006C 69            143          MOV     L,C
006D 0604          144          MVI     B,P       ; LOOP COUNT
006F 37            145          STC               ; TWO'S COMPL INCR
                   146 NEGLP:
0070 7E            147          MOV     A,M
0071 4F            148          MOV     C,A       ; SAVE MSB FOR SIGN
                   149
0072 2F            150          CMA
0073 CE00          151          ACI     0
0075 77            152          MOV     M,A       ;(H,L)=NOT(H,L)+CARRY
0076 23            153          INX     H
0077 05            154          DCR     B         ; BYTES LEFT ?
0078 C27000    C   155          JNZ     NEGLP     ;   YES
007B D8            156          RC                ; IF CARRY, NO OVERFLOW
007C A9            157          XRA     C         ; OLD SIGN = NEW SIGN ?
007D F8            158          RM                ;   NO, OK
007E C31E00    C   159          JMP     FQFERH    ;   YES, OVERFLOW
                   160 ;
                   161 ;*******************************************
                   162 ;
                   163          PUBLIC  FQFSET,FQFCLR
                   164 FQFSET:          ; SET UP FLOATING POINT REGISTER
                   165 ; INPUTS:        STACK = ADDRESS OF REGISTER AREA
                   166 ; DESTROYS:      B,C,H,L
                   167 ;
0081 E1            168          POP     H         ; H,L = RETURN ADDRESS
0082 E3            169          XTHL              ; (H,L) = ACC
0083 44            170          MOV     B,H
0084 4D            171          MOV     C,L       ; (B,C) = ACC
                   172 ;
                   173 FQFCLR:          ; CLEAR ACC,EXT,AUX
                   174 ;
0085 CD3C00    C   175          CALL    SAVE
0088 AF            176          XRA     A
0089 160C          177          MVI     D,3*P
                   178 SETLP:
008B 02            179          STAX    B
008C 03            180          INX     B
008D 15            181          DCR     D
008E C28B00    C   182          JNZ     SETLP
0091 C9            183          RET
                   184 ;
                   185 ;********************************************
                   186 ;
                   187          PUBLIC  FQFCM4
                   188 FQFCM4:          ; COMPARE OPERATION
```

```
 LOC  OBJ           SEQ          SOURCE STATEMENT

                    189 ; OUTPUTS:      A = RESULT OF COMPARISON
                    190 ;          80: ACC = OP
                    191 ;          40: ACC > OP
                    192 ;          20: ACC < OP
                    193 ;
 0092 C5            194            PUSH    B
 0093 D5            195            PUSH    D
 0094 E5            196            PUSH    H
                    197            REPT    P-1
                    198            INX     D
                    199            ENDM
 0095 13            200+           INX     D
 0096 13            201+           INX     D
 0097 13            202+           INX     D
 0098 210300        203            LXI     H,3
 009B 09            204            DAD     B       ; (H,L) = ACC.MSB
 009C 0604          205            MVI     B,P     ; B = LOOP COUNTER
 009E 7E            206            MOV     A,M
 009F A7            207            ANA     A       ; ACC < 0 ?
 00A0 F2B800   C    208            JP      ACCGE   ;   NO
 00A3 1A            209            LDAX    D
 00A4 A7            210            ANA     A       ; OP < 0 ?
 00A5 3E20          211            MVI     A,20H
 00A7 F2CD00   C    212            JP      COMRET  ;   NO
 00AA CDD100   C    213            CALL    COMPLP
 00AD E1            214            POP     H
 00AE D1            215            POP     D
 00AF C1            216            POP     B
 00B0 3E80          217            MVI     A,80H   ; ACC = OP ?
 00B2 C8            218            RZ              ;   YES
 00B3 3E40          219            MVI     A,40H   ; ACC > OP ?
                    220
 00B5 D0            221            RNC             ;   YES
 00B6 0F            222            RRC             ; ACC < OP (MUST BE)
 00B7 C9            223            RET
                    224 ACCGE:
 00B8 1A            225            LDAX    D
 00B9 A7            226            ANA     A       ; OP < 0 ?
 00BA 3E40          227            MVI     A,40H
 00BC FACD00   C    228            JM      COMRET
 00BF CDD100   C    229            CALL    COMPLP
 00C2 E1            230            POP     H
 00C3 D1            231            POP     D
 00C4 C1            232            POP     B
 00C5 3E80          233            MVI     A,80H   ; ACC > OP ?
 00C7 C8            234            RZ              ;   YES
 00C8 3E40          235            MVI     A,40H   ; ACC > OP
 00CA D8            236            RC              ;   YES
 00CB 0F            237            RRC             ; ACC < OP (MUST BE)
```

```
LOC  OBJ          SEQ        SOURCE STATEMENT

00CC C9            238         RET
                   239 ;
                   240 COMRET:
00CD E1            241         POP     H
00CE D1            242         POP     D
00CF C1            243         POP     B
00D0 C9            244         RET
                   245 ;
                   246 COMPLP:
00D1 1A            247         LDAX    D
00D2 BE            248         CMP     M
00D3 C0            249         RNZ
00D4 05            250         DCR     B
00D5 C8            251         RZ
00D6 1B            252         DCX     D
00D7 2B            253         DCX     H
00D8 C3D100    C   254         JMP     COMPLP
                   255 ;
                   256 ;*****************************************
                   257 ;
                   258         PUBLIC  FQFLD4,FQFFLM
                   259 FQFLD4:         ; LOAD ACC FROM OP
                   260 FQFFLM:         ; CONVERT 32-BIT INTEGER
                   261 ;
00DB CD3C00    C   262         CALL    SAVE
00DE 2604          263         MVI     H,P
                   264 LOADLP:
00E0 1A            265         LDAX    D
00E1 02            266         STAX    B
00E2 25            267         DCR     H
00E3 C8            268         RZ
00E4 13            269         INX     D
00E5 03            270         INX     B
00E6 C3E000    C   271         JMP     LOADLP
                   272 ;
                   273 ;*******************************************
                   274 ;
                   275         PUBLIC  FQFST4
                   276 FQFST4:         ; STORE FROM ACC TO OP
                   277 ;
00E9 CD3C00    C   278         CALL    SAVE
00EC 2604          279         MVI     H,P
                   280 STORLP:
00EE 0A            281         LDAX    B
00EF 12            282         STAX    D
00F0 25            283         DCR     H
00F1 C8            284         RZ
00F2 03            285         INX     B
00F3 13            286         INX     D
```

```
LOC  OBJ            SOURCE STATEMENT

00F4 C3EE00    C            JMP     STORLP
               ;
               ;*******************************************
               ;
                            PUBLIC  FQFML4,FQFMP4
               FQFML4:              ; MULTIPLY OPERATION
               FQFMP4:              ; SYNONYM
               ; CALLS:             MDAUX,EXTO,FQFNEG,FQFAD4
               ;
00F7 CD3C00    C            CALL    SAVE
00FA 210300                 LXI     H,P-1
00FD 09                     DAD     B
00FE 7E                     MOV     A,M     ; MSB OF ACC
00FF A7                     ANA     A       ; ACC < 0 ?
0100 FC8E01    C            CM      MDAUX   ;   YES, COMPLEMENT
0103 FC6800    C            CM      FQFNEG  ;     ... ACC & OP
0106 CDAC01    C            CALL    EXTO
0109 3E21                   MVI     A,P*8+1 ; P*8 BITS + SIGN
               ;
               MLOOP:               ; MAJOR MULTIPLY LOOP
010B F5                     PUSH    PSW     ; SAVE LOOP COUNT
010C C5                     PUSH    B       ; SAVE EXT.LSB ADDR
010D E5                     PUSH    H       ; ...  EXT.MSB
010E 0608                   MVI     B,2*P   ; SHIFT COUNT
0110 7E                     MOV     A,M
0111 17                     RAL             ; EXTEND SIGN
               MSLP:                ; SHIFT EXT & ACC RIGHT
0112 7E                     MOV     A,M
0113 1F                     RAR
0114 77                     MOV     M,A
0115 2B                     DCX     H
0116 05                     DCR     B
0117 C21201    C            JNZ     MSLP
011A E1                     POP     H
011B C1                     POP     B       ; RESTORE EXT ADDR
011C DC0000    C            CC      FQFAD4  ; ADD IF CARRY FROM LSB
011F F1                     POP     PSW
0120 3D                     DCR     A       ; LOOP COUNT
0121 C20B01    C            JNZ     MLOOP
               ; CHECK FOR OVERFLOW
0124 60                     MOV     H,B
0125 69                     MOV     L,C
0126 2B                     DCX     H       ; (H,L) = MSB ACC
0127 7E                     MOV     A,M
0128 07                     RLC
0129 E601                   ANI     1       ; A.0 = SIGN ACC
012B 0604                   MVI     B,4     ; LOOP COUNT
               MOFLP:   ; ADD SIGN TO EXT
012D 23                     INX     H
```

AD-A106 176 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/G 20/8
FEASIBILITY OF INTERFACING A MICROCOMPUTER WITH A MULTICHANNEL --ETC(U)
MAR 81 D CLARKE
UNCLASSIFIED AFIT/GNE/PH/81M-2 NL

2 OF 2
AD
A106176

END
DATE
FILMED
11 8
DTIC

```
 LOC  OBJ         SEQ        SOURCE STATEMENT

012E 8E            336        ADC    M
012F C21E00   C    337        JNZ    FQFERH  ; EXT + SIGN(ACC) <> 0
0132 05            338        DCR    B
0133 C22D01   C    339        JNZ    MOFLP
0136 C9            340        RET
                   341 ;
                   342 ;******************************************
                   343 ;
                   344        PUBLIC FQFDV4
                   345 FQFDV4:           ; DIVIDE OPERATION
                   346 ; OUTPUTS:        ACC = QUOTIENT
                   347 ;                 EXT = REMAINDER
0137 CD3C00   C    348        CALL   SAVE
013A 210300        349        LXI    H,P-1
013D 09            350        DAD    B       ; (H,L) = MSB OF ACC
013E 7E            351        MOV    A,M
013F 210300        352        LXI    H,P-1
0142 19            353        DAD    D       ; (H,L) = MSB OF OP
0143 A7            354        ANA    A       ; ACC < 0 ?
0144 FC6800   C    355        CM     FQFNEG  ;   YES, MAKE +
0147 17            356        RAL
0148 7E            357        MOV    A,M
0149 F5            358        PUSH   PSW     ; SAVE SIGNS FOR END
014A A7            359        ANA    A       ; OP < 0 ?
014B FC8E01   C    360        CM     MDAUX   ;   YES, MAKE +
014E C5            361        PUSH   B       ; SAVE ACC ADDR
014F CDAC01   C    362        CALL   EXT0
0152 E1            363        POP    H       ; (H,L) = ACC
0153 3E20          364        MVI    A,P*8   ; A = LOOP COUNT
                   365 ;
                   366 DLOOP:            ; MAJOR DIVIDE LOOP
0155 F5            367        PUSH   PSW     ; SAVE LOOP COUNT
0156 E5            368        PUSH   H       ;  ... ADDR ACC
0157 C5            369        PUSH   B       ;        ... EXT
0158 0608          370        MVI    B,2*P   ; SHIFT COUNT
015A A7            371        ANA    A       ; CLEAR CARRY
                   372 DSLP:             ; SHIFT EXT & ACC LEFT
015B 7E            373        MOV    A,M
015C 17            374        RAL
015D 77            375        MOV    M,A
015E 23            376        INX    H
015F 05            377        DCR    B
0160 C25B01   C    378        JNZ    DSLP
0163 C1            379        POP    B       ; (B,C) = EXT
                   380 ; TRY SUBTRACTION, IF POSITIVE, LEAVE IT
0164 CD4900   C    381        CALL   FQFSB4
0167 210300        382        LXI    H,P-1
016A 09            383        DAD    B       ; (H,L) = EXT.MSB
016B 7E            384        MOV    A,M
```

```
LOC  OBJ           SEQ          SOURCE STATEMENT

016C E1             385          POP     H         ; (H,L) = ACC
016D A7             386          ANA     A         ; EXT > 0 ?
016E FC0000    C    387          CM      FQFAD4    ;   NO, ADD OP BACK
0171 FA7501    C    388          JM      DECR
0174 34             389          INR     M         ;   YES, QUOTIENT = 1
                    390 DECR:                      ; DECREMENT LOOP COUNT
0175 F1             391          POP     PSW
0176 3D             392          DCR     A
0177 C25501    C    393          JNZ     DLOOP
                    394 ; FIX UP SIGNS OF QUOTIENT & REMAINDER
017A 0B             395          DCX     B
017B 0A             396          LDAX    B
017C A7             397          ANA     A
017D FA1E00    C    398          JM      FQFERH
0180 F1             399          POP     PSW
0181 03             400          INX     B         ; (B,C) = REMAINDER
0182 DC6800    C    401          CC      FQFNEG    ; ORIGINAL ACC < 0 ?
0185 44             402          MOV     B,H
0186 4D             403          MOV     C,L
0187 1F             404          RAR
0188 E6C0           405          ANI     11000000B
018A E46800    C    406          CPO     FQFNEG
018D C9             407          RET
                    408 ;
                    409 ;****************************************
                    410 ;
                    411 MDAUX:                     ; MOVE -OP TO AUX REGISTER
                    412 ;
018E F5             413          PUSH    PSW
018F 210800         414          LXI     H,2*P
0192 09             415          DAD     B
0193 C5             416          PUSH    B
0194 E5             417          PUSH    H
                    418          REPT    P-1
                    419          LDAX    D
                    420          MOV     M,A
                    421          INX     H
                    422          INX     D
                    423          ENDM
0195 1A             424+         LDAX    D
0196 77             425+         MOV     M,A
0197 23             426+         INX     H
0198 13             427+         INX     D
0199 1A             428+         LDAX    D
019A 77             429+         MOV     M,A
019B 23             430+         INX     H
019C 13             431+         INX     D
019D 1A             432+         LDAX    D
019E 77             433+         MOV     M,A
```

```
LOC  OBJ          SEQ         SOURCE STATEMENT

019F 23           434+         INX    H
01A0 13           435+         INX    D
01A1 1A           436          LDAX   D
01A2 77           437          MOV    M,A      ; MOVE OP INTO AUX
01A3 C1           438          POP    B
01A4 CD6800   C   439          CALL   FQFNEG   ; OP = -OP
01A7 50           440          MOV    D,B
01A8 59           441          MOV    E,C
01A9 C1           442          POP    B        ; (B,C) = ACC
01AA F1           443          POP    PSW
01AB C9           444          RET
                  445 ;
                  446 ;*****************************************
                  447 ;
                  448 EXTO:               ; SET EXT REGISTER TO ZERO
                  449 ;
01AC 210400       450          LXI    H,P
01AF 09           451          DAD    B        ; (H,L) = EXT
01B0 44           452          MOV    B,H
01B1 4D           453          MOV    C,L      ; (B,C) = EXT
01B2 AF           454          XRA    A
01B3 77           455          MOV    M,A
01B4 23           456          INX    H
01B5 77           457          MOV    M,A      ; EXT = 0
01B6 23           458          INX    H
01B7 77           459          MOV    M,A
01B8 23           460          INX    H
01B9 77           461          MOV    M,A      ; (H,L) = EXT.MSB
01BA C9           462          RET
                  463 ;
                  464 ;*****************************************
                  465 ;
                  466          PUBLIC FQFABS
                  467 FQFABS:             ; ABSOLUTE VALUE OPERATION
                  468 ; CALLS:            SAVE
01BB CD3C00   C   469          CALL   SAVE
01BE 210300       470          LXI    H,P-1
01C1 09           471          DAD    B        ; (H,L) = MSB OF ACC
01C2 7E           472          MOV    A,M
01C3 A7           473          ANA    A        ; ACC < 0 ?
01C4 FC6800   C   474          CM     FQFNEG   ;   YES, MAKE POSITIVE
01C7 C9           475          RET
                  476 ;
                  477 ;*****************************************
                  478 ;
                  479          PUBLIC FMUL10
                  480 FMUL10:             ; MULTIPLY BY 10 OPERATION
                  481 ; CALLS:            SAVE,FQFAD4
01C8 CD3C00   C   482          CALL   SAVE
```

```
  LOC  OBJ         SEQ         SOURCE STATEMENT

  01CB CDE401   C   483          CALL    SHIFT    ; TIMES 2
  01CE C5           484          PUSH    B
  01CF 2604         485          MVI     H,P      ; LOOP COUNT
                    486 FM10LP:          ; MOVE *2 TO EXT
  01D1 0A           487          LDAX    B
  01D2 12           488          STAX    D
  01D3 03           489          INX     B
  01D4 13           490          INX     D
  01D5 25           491          DCR     H
  01D6 C2D101   C   492          JNZ     FM10LP
  01D9 C1           493          POP     B
  01DA CDE401   C   494          CALL    SHIFT    ; TIMES 4
  01DD CDE401   C   495          CALL    SHIFT    ; TIMES 8
  01E0 CD0000   C   496          CALL    FQFAD4   ; ACC = *8 + *2
  01E3 C9           497          RET
                    498 ;
                    499 SHIFT:           ; SHIFT ACC LEFT LOGICAL
                    500 ; OUTPUTS:       D,E = B,C + P
                    501 ; DESTROYS:      A,H
  01E4 50           502          MOV     D,B
  01E5 59           503          MOV     E,C
  01E6 2604         504          MVI     H,P      ; LOOP COUNT
  01E8 A7           505          ANA     A        ; START, CARRY = 0
                    506 SHFLP:
  01E9 1A           507          LDAX    D        ; SHIFT LEFT
  01EA 17           508          RAL              ;  ... ONE BYTE
  01EB 12           509          STAX    D        ;  ... AT A TIME
  01EC 13           510          INX     D
  01ED 25           511          DCR     H
  01EE C2E901   C   512          JNZ     SHFLP
  01F1 DA1E00   C   513          JC      FQFERH   ; CARRY = OVERFLOW
  01F4 C9           514          RET
                    515 ;
                    516 ;****************************************
                    517 ;
                    518          END

PUBLIC SYMBOLS
FMUL10 C 01C8    FQFABS C 01BB    FQFAD4 C 0000    FQFCLR C 00E5
FQFCM4 C 0092    FQFDV4 C 0137    FQFERH C 001E    FQFFLM C 00DB
FQFLD4 C 00DB    FQFML4 C 00F7    FQFMP4 C 00F7    FQFNEG C 0068
FQFSB4 C 0049    FQFSET C 0081    FQFST4 C 00E9

EXTERNAL SYMBOLS
CRLF   E 0000    OUTSTR E 0000

USER SYMBOLS
ACCGE  C 00B8    ADDLP  C 0008    COMPLP C 00D1    COMRET C 00CC
CRLF   E 0000    DECR   C 0175    DLOOP  C 0155    DSLP   C 015B
```

```
ERRMSG C 0028   EXTO   C 01AC   FM10LP C 01D1   FMUL10 C 01C8
FQFABS C 01BB   FQFAD4 C 0000   FQFCLR C 0085   FQFCM4 C 0092
FQFDV4 C 0137   FQFERH C 001E   FQFFLM C 00DB   FQFLD4 C 00DB
FQFML4 C 00F7   FQFMP4 C 00F7   FQFNEG C 0068   FQFSB4 C 0049
FQFSET C 0081   FQFST4 C 00E9   JUMP   C 0048   LOADLP C 00E0
MDAUX  C 018E   MLOOP  C 010B   MOFLP  C 012D   MSLP   C 0112
NEGLP  C 0070   OUTSTR E 0000   P      A 0004   SAVE   C 003C
SETLP  C 008B   SHFLP  C 01E9   SHIFT  C 01E4   STORLP C 00EE
SUBLP  C 0052

ASSEMBLY COMPLETE,  NO ERRORS
```

```
ASM :F1:MONIT.SRC PAGELENGTH(60)

ISIS-II 8080/8085 MACRO ASSEMBLER, V2.0          MONIT    PAGE    1
MONIT - REPLACE SBC MONITOR

 LOC  OBJ          SEQ          SOURCE STATEMENT

                     1 $MACROFILE PAGEWIDTH(79)
                     2 $TITLE('MONIT - REPLACE SBC MONITOR')
                     3 ;
                     4 ; 13 JAN 81
                     5 ;
                     6         NAME    MONIT
                     7         EXTRN   OUTSTR,CRLF,GAMMA
                     8         PUBLIC  CI,CO
                     9 ;
                    10         CSEG
                    11 ;
                    12 ; SYSTEM EQUATES
                    13 ;
 0000               14 MDS     SET     0       ; MDS OR SBC SWITCH
                    15         IF      MDS
                    16 USART   EQU     0F6H    ; MDS PORT ASSIGNMENTS
                    17 ICCP    EQU     0FCH
                    18         ELSE
 00EC               19 USART   EQU     0ECH    ; RS232 PORT
 00DC               20 TIMER   EQU     0DCH    ; COUNTER/TIMER PORT
 00DA               21 ICCP    EQU     0DAH    ; INTERUPT CONTROL PORT
 00E7               22 OUTCTL  EQU     0E7H    ; CONTROL E4,E5,E6
 00EB               23 INCTL   EQU     0EBH    ; CONTROL E8,E9,EA
 00E6               24 OUTBYT  EQU     0E6H    ; OUTPUT TO ALL MCA LINES
                    25 ;
 0000               26         ORG     0       ; RESET
                    27 ZERO:   ; RELOCATABLE ZERO SYMBOL
 0000 AF            28         XRA     A
 0001 47            29         MOV     B,A     ; FROM ADDRESS = 0
 0002 4F            30         MOV     C,A
 0003 2680          31         MVI     H,80H   ; TO ADDRESS = 8000H
 0005 C30F00        32         JMP     MOVE-1
                    33 ;
 0008               34         ORG     8       ; RESTART 1 (BREAKPOINT)
 0008 F3            35         DI
 0009 CDC001    C   36         CALL    REGSV
 000C C33501    C   37         JMP     BREAK
                    38 ;
 000F 6F            39         MOV     L,A     ; FINISH ADDRESS
 0010               40 MOVE    EQU     $-ZERO  ; ABSOLUTE ADDRESS
                    41 ; MOVE PROGRAM FROM EPROM TO RAM
 0010 0A            42         LDAX    B
 0011 77            43         MOV     M,A     ; MOVE ONE BYTE
 0012 03            44         INX     B
 0013 23            45         INX     H
 0014 78            46         MOV     A,B
```

```
LOC  OBJ          SEQ        SOURCE STATEMENT

0015 FE10          47        CPI     10H      ; DONE ?
0017 CA1D00   C    48        JZ      INIT     ;   YES
001A C31000        49        JMP     MOVE     ;   NO, LOOP
                   50 ;
                   51 ; THE PRECEDING CODE RUNS AT ORIGIN 0
                   52 ; THE SUCCEEDING CODE RUNS AT ORIGIN 8000H
                   53 ;
                   54 INIT:               ; INITIALIZE AFTER RESET
                   55                     ; ASSUMES HARDWARE RESET
                   56 ;
                   57 ; SERIAL I/O INITIALIZATION
001D 3E4E          58        MVI     A,01001110B ; 1 STOP,NO PARITY,
                   59                            ; 8 BIT,16*BAUD
001F D3ED          60        OUT     USART+1 ; MODE INSTRUCTION
0021 3E27          61        MVI     A,00100111B ; RTS,RXE,*DTR,TXE
0023 D3ED          62        OUT     USART+1 ; COMMAND INSTRUCTION
0025 31D03F        63        LXI     SP,MSTAK
0028 3EB6          64        MVI     A,10110110B ; CTR2,BOTH,MODE3,BIN
002A D3DF          65        OUT     TIMER+3 ; CONTROL WORD
002C 21E000        66        LXI     H,32*7  ; 300 BAUD
002F 7D            67        MOV     A,L
0030 D3DE          68        OUT     TIMER+2 ; COUNTER 2 LSB
0032 7C            69        MOV     A,H
0033 D3DE          70        OUT     TIMER+2 ; COUNTER 2 MSB
0035 3E37          71        MVI     A,00110111B ; RTS,RESET,RXE,
                   72                            ; *DTR,TXE
0037 D3ED          73        OUT     USART+1 ; COMMAND INSTRUCTION
                   74 ;
                   75 ; PARALLEL I/O INITIALIZATION
0039 3E90          76        MVI     A,10010000B ; MODE=0 1-I 2-O 3-O
003B D3E7          77        OUT     OUTCTL
003D 3E9B          78        MVI     A,10011011B ; MODE=0 4-I 5-I 6-I
003F D3EB          79        OUT     INCTL
0041 AF            80        XRA     A
0042 D3E6          81        OUT     OUTBYT  ; ALL LINES HIGH
                   82 ;
                   83 ; INITIALIZE INTERUPT CONTROLLER
0044 3E08          84        MVI     A,8
0046 110C01   C    85        LXI     D,INTRPT ; ADDRESS TO JUMP TO
0049 21E03F        86        LXI     H,JTABLE ; PLACE TO PUT JUMPS
                   87 INI05:
004C 36C3          88        MVI     M,0C3H  ; JMP OPCODE
004E 23            89        INX     H
004F 73            90        MOV     M,E     ; ADDRESS LSB
0050 23            91        INX     H
0051 72            92        MOV     M,D     ; ... MSB
0052 23            93        INX     H
0053 23            94        INX     H
0054 3D            95        DCR     A
```

```
 LOC  OBJ         SEQ        SOURCE STATEMENT

0055 C24C00   C    96          JNZ     INI05   ; MOVE JUMP TABLE TO DATA
0058 3EF6          97          MVI     A,(LOW JTABLE) +10110B
                   98                          ; A7-5,4 BYTE,SINGLE
005A D3DA          99          OUT     ICCP    ; INTERUPT CONTROL WORD 1
005C 3E3F         100          MVI     A,HIGH JTABLE ; A15-8
005E D3DB         101          OUT     ICCP+1  ; INTERUPT CONTROL WORD 2
0060 AF           102          XRA     A
0061 D3DB         103          OUT     ICCP+1  ; NO INTERUPTS MASKED
                  104          ENDIF
                  105 START:
0063 210000   E   106          LXI     H,GAMMA ; START ADDRESS OF MAIN
0066 22D83F       107          SHLD    PCSAVE  ; ... ROUTINE IS DEFAULT
0069 21C03F       108          LXI     H,MSTAK-16
006C F9           109          SPHL
006D 22DA3F       110          SHLD    SPSAVE  ; DEFAULT STACK POINTER
                  111 ; SEND SIGNON MESSAGE
0070 CD0000   E   112          CALL    CRLF
0073 219F00   C   113          LXI     H,SGNON
0076 CD0000   E   114          CALL    OUTSTR
                  115 ;
                  116 ;*******************************************
                  117 ;
                  118 GETCMD:              ; GET NEXT COMMAND
                  119 ;
0079 31D03F       120          LXI     SP,MSTAK ; REINITIALIZE STACK
007C CD0000   E   121          CALL    CRLF
007F 0E2E         122          MVI     C,'.'   ; PROMPT FOR COMMAND
0081 CD5C01   C   123          CALL    CO
0084 CD5001   C   124          CALL    CI
0087 4F           125          MOV     C,A
0088 CD5C01   C   126          CALL    CO      ; ECHO THE COMMAND
008B 79           127          MOV     A,C
008C 010400       128          LXI     B,4     ; B,C = INCR/COMMAND
008F 21B100   C   129          LXI     H,CTABLE
                  130 GTC05:
0092 BE           131          CMP     M       ; THIS COMMAND ?
0093 CA9D00   C   132          JZ      GTC10   ;   YES
0096 DA6701   C   133          JC      ERROR   ;   NEVER
0099 09           134          DAD     B       ;   MAYBE NEXT
009A C39200   C   135          JMP     GTC05
                  136 GTC10:
009D 23           137          INX     H
009E E9           138          PCHL
009F 4D494352     139 SGNON:   DB      'MICRO GAMMA  V1.0',0
00A3 4F204741
00A7 4D4D4120
00AB 2056312E
00AF 30
00B0 00
```

```
 LOC  OBJ         SEQ        SOURCE STATEMENT

                  140 CTABLE:             ; COMMANDS IN ALPHA ORDER
 00B1 44          141         DB       'D'
 00B2 C3BE00   C  142         JMP      DCMD
 00B5 47          143         DB       'G'
 00B6 C3DD00   C  144         JMP      GCMD
 00B9 53          145         DB       'S'
 00BA C3E900   C  146         JMP      SCMD
 00BD FF          147         DB       0FFH     ; END OF COMMANDS
                  148 ;
                  149 ;****************************************
                  150 ;
                  151 DCMD:               ; DISPLAY COMMAND
 00BE CD6F01   C  152         CALL     GETHX    ; GET START ADDRESS
 00C1 D26701   C  153         JNC      ERROR    ;   NONE
 00C4 CD0000   E  154         CALL     CRLF
 00C7 CD4701   C  155         CALL     ADRD     ; PRINT ADDRESS
 00CA 1610        156         MVI      D,16     ;  ... AND 16 DATA BYTES
                  157 DCM10:
 00CC 0E20        158         MVI      C,' '    ; BLANK AS SEPARATOR
 00CE CD5C01   C  159         CALL     CO
 00D1 7E          160         MOV      A,M
 00D2 CDAB01   C  161         CALL     NMOUT    ; PRINT ONE DATA BYTE
 00D5 23          162         INX      H
 00D6 15          163         DCR      D
 00D7 C2CC00   C  164         JNZ      DCM10
 00DA C37900   C  165         JMP      GETCMD
                  166 ;
                  167 ;******************************************
                  168 ;
                  169 GCMD:               ; GO COMMAND
 00DD CD6F01   C  170         CALL     GETHX    ; GET ADDR TO START
 00E0 D2D901   C  171         JNC      RSTTF    ;   NONE, USE LAST PC
 00E3 22D83F      172         SHLD     PCSAVE
 00E6 C3D901   C  173         JMP      RSTTF
                  174 ;
                  175 ;******************************************
                  176 ;
                  177 SCMD:               ; SET MEMORY COMMAND
 00E9 CD6F01   C  178         CALL     GETHX
 00EC D26701   C  179         JNC      ERROR    ; NO START ADDRESS
                  180 SCM05:
 00EF 7A          181         MOV      A,D
 00F0 FE20        182         CPI      ' '      ; DELIM = ' ' ?
 00F2 C27900   C  183         JNZ      GETCMD   ;   NO, DONE
                  184 SCM10:
 00F5 7E          185         MOV      A,M
 00F6 CDAB01   C  186         CALL     NMOUT    ; PRINT CURRENT VALUE
 00F9 0E2D        187         MVI      C,'-'
 00FB CD5C01   C  188         CALL     CO
```

```
LOC  OBJ          SEQ         SOURCE STATEMENT

00FE E5            189          PUSH    H
00FF CD6F01     C  190          CALL    GETHX    ; GET NEW VALUE
0102 4D            191          MOV     C,L
0103 E1            192          POP     H
0104 D20801     C  193          JNC     SCM15    ;  WAS THERE ONE ?
0107 71            194          MOV     M,C      ;    YES
                   195 SCM15:
0108 23            196          INX     H        ; NEXT BYTE
0109 C3EF00     C  197          JMP     SCM05
                   198 ;
                   199 ;*******************************************
                   200 ;
                   201 INTRPT:                  ; PROCESS INTERUPT
010C CDC001     C  202          CALL    REGSV    ; SAVE ALL
010F 211901     C  203          LXI     H,INTMSG
0112 3E20          204          MVI     A,20H    ; UNSPECIFIC END OF INT
0114 D3DA          205          OUT     ICCP
0116 C33801     C  206          JMP     ADROUT   ; PRINT ADDRESS
0119 494E5445      207 INTMSG: DB       'INTERUPT AT ',0
011D 52555054
0121 20415420
0125 00
0126 42524541      208 BRKMSG: DB       'BREAKPOINT AT ',0
012A 4B504F49
012E 4E542041
0132 5420
0134 00
                   209 BREAK:
0135 212601     C  210          LXI     H,BRKMSG
                   211 ;
                   212 ;*******************************************
                   213 ;
                   214 ADROUT:                  ; PRINT ADDRESS OF BREAK/INTERUPT
0138 CD0000     E  215          CALL    CRLF
013B CD0000     E  216          CALL    OUTSTR
013E 2AD83F        217          LHLD    PCSAVE
0141 CD4701     C  218          CALL    ADRD
0144 C37900     C  219          JMP     GETCMD
                   220
                   221
                   222 ADRD:                    ; PRINT ADDRESS IN H,L
0147 7C            223          MOV     A,H
0148 CDAB01     C  224          CALL    NMOUT
014B 7D            225          MOV     A,L
014C CDAB01     C  226          CALL    NMOUT
014F C9            227          RET
                   228
                   229
                   230 CI:                      ; GET ONE BYTE FROM CONSOLE
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

0150 DBED          231          IN      USART+1
0152 E602          232          ANI     10B       ; RCV BUFFER FULL ?
0154 CA5001   C    233          JZ      CI        ;   NO
0157 DBEC          234          IN      USART     ; GET CHAR
0159 E67F          235          ANI     7FH       ; GET RID OF PARITY
015B C9            236          RET
                   237
                   238
                   239 CO:                ; PRINT ONE BYTE AT CONSOLE
015C DBED          240          IN      USART+1
015E E601          241          ANI     1B        ; XMIT BUFFER EMPTY ?
0160 CA5C01   C    242          JZ      CO        ;   NO
0163 79            243          MOV     A,C
0164 D3EC          244          OUT     USART     ; PUT CHAR
0166 C9            245          RET
                   246
                   247
                   248 ERROR:             ; TELL USER HE GOOFED
0167 0E23          249          MVI     C,'#'
0169 CD5C01   C    250          CALL    CO
016C C37900   C    251          JMP     GETCMD
                   252
                   253
                   254 GETHX:             ; GET 4 HEX DIGITS
016F 210000        255          LXI     H,0       ; H,L = OUTPUT = 0
0172 1E00          256          MVI     E,0       ; FLAG = NOTHING YET
                   257 GHX05:
0174 CD5001   C    258          CALL    CI        ; GET CHAR
0177 4F            259          MOV     C,A
0178 CD5C01   C    260          CALL    CO        ;  ... ECHO
017B CDF001   C    261          CALL    VALDL     ; DELIMETER ?
017E C28501   C    262          JNZ     GHX10     ;   NO
0181 51            263          MOV     D,C       ; D = DELIMETER
0182 7B            264          MOV     A,E
0183 07            265          RLC     ; CARRY = ONE OR MORE DIGITS
0184 C9            266          RET
                   267 GHX10:
0185 CD9701   C    268          CALL    CNVBN
0188 1EFF          269          MVI     E,0FFH    ; FLAG = SOMETHING
018A 29            270          DAD     H         ; *2
018B 29            271          DAD     H         ; *4
018C 29            272          DAD     H         ; *8
018D 29            273          DAD     H         ; H,L = H,L * 16
018E 85            274          ADD     L
018F 6F            275          MOV     L,A
0190 D27401   C    276          JNC     GHX05
0193 24            277          INR     H         ; H,L = H,L + A
0194 C37401   C    278          JMP     GHX05
                   279
```

```
LOC  OBJ          SEQ        SOURCE STATEMENT

                  280
                  281 CNVBN:            ; CONVERT HEX ASCII --> BINARY
0197 79           282         MOV     A,C
0198 D630         283         SUI     '0'     ; < 0 ?
019A DA6701    C  284         JC      ERROR   ;   YES
019D FE0A         285         CPI     10      ; < 10 ?
019F D8           286         RC              ;   YES, 0-9
01A0 C6F9         287         ADI     '0'-'A'+10 ; < A ?
01A2 D26701    C  288         JNC     ERROR      ;   YES
01A5 FE10         289         CPI     16      ; > F
01A7 D8           290         RC              ;   NO, A-F
01A8 C36701    C  291         JMP     ERROR   ;   YES
                  292
                  293
                  294 NMOUT:            ; PRINT ONE BYTE AS 2 HEX DIGITS
01AB F5           295         PUSH    PSW
                  296         REPT    4
                  297         RRC
                  298         ENDM
01AC 0F           299+        RRC
01AD 0F           300+        RRC
01AE 0F           301+        RRC
01AF 0F           302+        RRC
01B0 CDB401    C  303         CALL    PRVAL   ; PRINT HIGH NIBBLE
01B3 F1           304         POP     PSW
                  305 PRVAL:            ; PRINT LOW NIBBLE AS HEX DIGIT
01B4 E60F         306         ANI     0FH
01B6 C690         307         ADI     90H     ; 90 <-- 0    9A <-- A
01B8 27           308         DAA             ; 90          100
01B9 CE40         309         ACI     40H     ; D0           41
01BB 27           310         DAA             ;130 = '0'    41 = 'A'
01BC 4F           311         MOV     C,A
01BD C35C01    C  312         JMP     CO
                  313
                  314
                  315 REGSV:            ; SAVE REGISTERS
01C0 22D63F       316         SHLD    HLSAVE  ; H,L
01C3 E1           317         POP     H       ; CALLER
01C4 E3           318         XTHL            ; INTERUPT/BREAK
01C5 22D83F       319         SHLD    PCSAVE
01C8 F5           320         PUSH    PSW     ; PRESERVE FLAGS
01C9 210400       321         LXI     H,4
01CC 39           322         DAD     SP      ; SP AT INTERUPT
01CD 22DA3F       323         SHLD    SPSAVE
01D0 F1           324         POP     PSW
01D1 E1           325         POP     H       ; CALLER
01D2 31D63F       326         LXI     SP,HLSAVE
01D5 F5           327         PUSH    PSW
01D6 C5           328         PUSH    B
```

```
 LOC  OBJ         SEQ         SOURCE STATEMENT

 01D7 D5           329        PUSH    D        ; NOW SP = MSTAK
 01D8 E9           330        PCHL             ; RETURN TO CALLER
                   331
                   332
                   333 RSTTF:          ; RESTORE REGISTERS AND START
 01D9 CD0000   E   334        CALL    CRLF
 01DC F3           335        DI
 01DD 31D03F       336        LXI     SP,MSTAK ; REGISTERS SAVED
 01E0 D1           337        POP     D        ;  ... ABOVE STACK
 01E1 C1           338        POP     B
 01E2 F1           339        POP     PSW
 01E3 2ADA3F       340        LHLD    SPSAVE
 01E6 F9           341        SPHL             ; STACK POINTER
 01E7 2AD83F       342        LHLD    PCSAVE   ; BREAK/INTERUPT POINT
 01EA E5           343        PUSH    H
 01EB 2AD63F       344        LHLD    HLSAVE   ; H,L
 01EE FB           345        EI
 01EF C9           346        RET
                   347
                   348
                   349 VALDL:          ; CHECK FOR VALID DELIMETER
 01F0 79           350        MOV     A,C
 01F1 FE20         351        CPI     ' '      ; BLANK ?
 01F3 C8           352        RZ               ;   YES
 01F4 FE0D         353        CPI     13       ; CARRIAGF RETURN ?
 01F6 C9           354        RET              ;   Z STATUS TELLS TALE
                   355 ;
                   356 ;****************************************
                   357 ;
                   358        ASEG
 3FD0              359        ORG     4000H-48
                   360 MSTAK:                  ; MONITOR STACK
 0006              361        DS      6        ; REGISTERS DE,BC,PSW
 0002              362 HLSAVE: DS     2        ;       HL
 0002              363 PCSAVE: DS     2        ; PROGRAM COUNTER
 0002              364 SPSAVE: DS     2        ; STACK POINTER
                   365
                   366
 3FE0              367        ORG     4000H-32
 0020              368 JTABLE: DS     32       ; INTERUPT JUMP TABLE
                   369
                   370
 0063          C   371        END     START    ; START FOR MDS USE

PUBLIC SYMBOLS
CI      C 0150    CO      C 015C

EXTERNAL SYMBOLS
CRLF    E 0000    GAMMA   E 0000    OUTSTR E 0000
```

USER SYMBOLS

ADRD	C	0147	ADROUT	C	0138	BREAK	C	0135	BRKMSG	C	0126
CI	C	0150	CNVBN	C	0197	CO	C	015C	CRLF	E	0000
CTABLE	C	00B1	DCM10	C	00CC	DCMD	C	00BE	ERROR	C	0167
GAMMA	E	0000	GCMD	C	00DD	GETCMD	C	0079	GETHX	C	016F
GHX05	C	0174	GHX10	C	0185	GTC05	C	0092	GTC10	C	009D
HLSAVE	A	3FD6	ICCP	A	00DA	INCTL	A	00EB	IN105	C	004C
INIT	C	001D	INTMSG	C	0119	INTRPT	C	010C	JTABLE	A	3FE0
MDS	A	0000	MOVE	A	0010	MSTAK	A	3FD0	NMOUT	C	01AB
OUTBYT	A	00E6	OUTCTL	A	00E7	OUTSTR	E	0000	PCSAVE	A	3FD8
PRVAL	C	01B4	REGSV	C	01C0	RSTTF	C	01D9	SCM05	C	00EF
SCM10	C	00F5	SCM15	C	0108	SCMD	C	00E9	SGNON	C	009F
SPSAVE	A	3FDA	START	C	0063	TIMER	A	00DC	USART	A	00EC
VALDL	C	01F0	ZERO	C	0000						

ASSEMBLY COMPLETE, NO ERRORS

```
ISIS-II FORTRAN-80 V1.0 COMPILATION OF PROGRAM UNIT PKPROC
OBJECT MODULE PLACED IN PKPROC.OBJ
COMPILER INVOKED BY:  :F1:FORT80 PKPROC.FOR PAGELENGTH(54)


  1      $PAGEWIDTH(73) DATE(17 NOV 80)
  2      $TITLE('PKPROC - PROCESS ONE PEAK')
  3              SUBROUTINE PKPROC(NPKL,NPKR,LASTR)
         C FIND CHANNEL NUMBER AND AREA OF PEAK
         C   FIND CHANNEL OF MAX COUNTS BETWEEN NPKL & NPKR
         C   FIND LOCAL MINIMUM ON BOTH SIDES OF MAXIMUM
         C   ASSUME TRAPEZOIDAL BACKGROUND
         C   COMPUTE NET PEAK AREA
         C   COMPUTE "EXACT" PEAK LOCATION BY FIRST MOMENT
         C
         C NOTE: MAX WIDTH OF PEAK ASSUMED TO BE 30 CHANNELS
         C
  4              COMMON NCHANL,NPEAKS,COEFF(8)
  5              COMMON CHANL(50),TOTAL(50),AREA(50),PKERGY(50),PGPS(50)
  6              COMMON TIME,COUNTS(4095)
         C
  7              NPEAKS=NPEAKS+1
  8              CALL OUTCHR('PEAK #')
  9              CALL OUTINT(NPEAKS,0)
 10              CALL OUTCHR(' BETWEEN ')
 11              CALL OUTINT(NPKL,0)
 12              CALL OUTCHR(' AND ')
 13              CALL OUTINT(NPKR,0)
 14              CALL CRLF
 15              IF(NPEAKS.EQ.50) CALL OUTCHR('*LIMIT* NO FURTHER PEAKS P
                    -ROCESSED')
 16              IF(NPEAKS.GT.50) RETURN
         C FIND CHANNEL WITH MAX COUNTS
 17              TEST=0
 18              DO 20 I=NPKL,NPKR
 19                IF(COUNTS(I).GT.TEST) THEN
 20                  TEST=COUNTS(I)
 21                  NPEAK=I
 22                ENDIF
 23      20        CONTINUE
 24              CALL OUTCHR('   MAX COUNT IN CHANNEL ')
 25              CALL OUTINT(NPEAK,0)
 26              CALL CRLF
         C FIND LOCAL MINIMA
 27              LIMIT=20
 28              IF(NPEAK-LIMIT.LT.1) LIMIT=NPEAK-2
 29              DO 100 I=1,LIMIT
 30                NLEFT=NPEAK-I
 31                IF(COUNTS(NLEFT).LE.COUNTS(NLEFT-1)) GO TO 120
 32      100       CONTINUE
```

```
            CALL OUTCHR('   LEFT LIMIT')
      120   LIMIT=10
            IF(NPEAK+LIMIT.GT.NCHANL) LIMIT=NCHANL-NPEAK-1
            DO 130 I=1,LIMIT
              NRIGHT=NPEAK+I
              IF(COUNTS(NRIGHT).LE.COUNTS(NRIGHT+1)) GO TO 140
      130     CONTINUE
            CALL OUTCHR('   RIGHT LIMIT')
      C SLOPE OF BACKGROUND (SCALED 1000)
      140   SLOPE=1000*(COUNTS(NRIGHT)-COUNTS(NLEFT))/(NRIGHT-NLEFT)
      C INTEGRATE UNDER PEAK
            CALL OUTCHR('   INTEGRATING ')
            CALL OUTINT(NLEFT,0)
            CALL OUTCHR(' TO ')
            CALL OUTINT(NRIGHT,0)
            CALL CRLF
            TOTAL(NPEAKS)=0
            AREA(NPEAKS)=0
            FIRSTM=0
            DO 200 I=NLEFT,NRIGHT
              TOTAL(NPEAKS)=TOTAL(NPEAKS)+COUNTS(I)
              ABOVE=COUNTS(I)-COUNTS(NLEFT)-SLOPE*(I-NLEFT)/1000
              AREA(NPEAKS)=AREA(NPEAKS)+ABOVE
              FIRSTM=FIRSTM+I*ABOVE
      200     CONTINUE
            CALL OUTCHR('     TOTAL COUNTS = ')
            CALL OUTFLT(TOTAL(NPEAKS),0)
            CALL CRLF
            CALL OUTCHR('     NET AREA = ')
            CALL OUTFLT(AREA(NPEAKS),0)
            IF(AREA(NPEAKS).LE.0) THEN
              CALL OUTCHR('    ** CAUTION **')
              CHANL(NPEAKS)=FLOAT(NPEAK)*10
            ELSE
              CHANL(NPEAKS)=10*FIRSTM/AREA(NPEAKS)
              CALL CRLF
              CALL OUTCHR('   PEAK AT CHANNEL ')
              CALL OUTFLT(CHANL(NPEAKS),100)
            ENDIF
            CALL CRLF
            RETURN
            END
```

MODULE INFORMATION:

```
     CODE AREA SIZE     = 055BH   1371D
     VARIABLE AREA SIZE = 0034H     52D
     MAXIMUM STACK SIZE = 000CH     12D
     86 LINES READ
```

```
ASM :F1:PKPROC.SRC PAGELENGTH(60) PAGEWIDTH(79)


ISIS-II 8080/8085 MACRO ASSEMBLER, V2.0        PKPROC    PAGE    1
PKPROC - AREA AND CHANNEL OF PEAK

 LOC  OBJ          SEQ         SOURCE STATEMENT

                     1 $MACROFILE NOCOND
                     2 $TITLE('PKPROC - AREA AND CHANNEL OF PEAK')
                     3 ;
                     4 ; 20 NOV 80
                     5 ;
                     6         NAME    PKPROC
                     7         EXTRN   CRLF,OUTSTR,OUTFLT,OUTINT,CO
                     8         EXTRN   FQFCLR,FQFLD4,FQFST4,FQFAD4
                     9         EXTRN   FQFSB4,FQFML4,FQFDV4,FQFCM4
                    10         EXTRN   FQFNEG
                    11         EXTRN   FQ0543,FQ0684,FQ0588,FQ0671
                    12         EXTRN   FQ0664,FQ0686
                    13 ;
                    14         CSEG
                    15 ;
                    16 ;***************************************
                    17 ;
                    18         PUBLIC  PKPROC
                    19 PKPROC:
                    20 ; PROCESS PARAMETER ADDRESSES
 0000 210500    D   21         LXI     H,LASTR+1H
 0003 72            22         MOV     M,D
 0004 2B            23         DCX     H
 0005 73            24         MOV     M,E
 0006 2B            25         DCX     H
 0007 70            26         MOV     M,B
 0008 2B            27         DCX     H
 0009 71            28         MOV     M,C
 000A 2B            29         DCX     H
 000B D1            30         POP     D          ; RETURN ADDRESS
 000C C1            31         POP     B
 000D 70            32         MOV     M,B
 000E 2B            33         DCX     H
 000F 71            34         MOV     M,C
 0010 D5            35         PUSH    D
                    36 ; NPEAKS = NPEAKS + 1
 0011 2A0290        37         LHLD    NPEAKS
 0014 23            38         INX     H
 0015 220290        39         SHLD    NPEAKS
                    40 ; PRINT CALLING ARGUMENTS (NPEAKS,NPKL,NPKR)
 0018 21CC02    C   41         LXI     H,MSG1
 001B CD0000    E   42         CALL    OUTSTR
 001E 11C802    C   43         LXI     D,ZERO
 0021 010290        44         LXI     B,NPEAKS
 0024 CD0000    E   45         CALL    OUTINT
 0027 21D302    C   46         LXI     H,MSG2
```

```
 LOC  OBJ          SEQ        SOURCE STATEMENT

0002A CD0000  E     47          CALL    OUTSTR
002D 2A0000   D     48          LHLD    NPKL
0030 E5             49          PUSH    H        ; 1 NPKL
0031 11C802   C     50          LXI     D,ZERO
0034 C1             51          POP     B        ; 1
0035 CD0000   E     52          CALL    OUTINT
0038 0E2D           53          MVI     C,'-'
003A CD0000   E     54          CALL    CO
003D 2A0200   D     55          LHLD    NPKR
0040 E5             56          PUSH    H        ; 1 NPKR
0041 11C802   C     57          LXI     D,ZERO
0044 C1             58          POP     B        ; 1
0045 CD0000   E     59          CALL    OUTINT
0048 CD0000   E     60          CALL    CRLF
                    61 ; CHECK WHETHER ROOM IN COMMON FOR THIS PEAK
004B 3A0290         62          LDA     NPEAKS   ; TOO MANY PEAKS ?
004E FE32           63          CPI     50
0050 DA5A00   C     64          JC      @1       ;   NO
0053 C0             65          RNZ              ;   YES
0054 21DB02   C     66          LXI     H,MSG4
0057 CD0000   E     67          CALL    OUTSTR   ;   LAST ONE
                    68 @1:
                    69 ; FIND CHANNEL WITH MAXIMUM COUNTS
005A 010600   D     70          LXI     B,@FAC
005D CD0000   E     71          CALL    FQFCLR   ; TEST = 0
                    72 ; DO 20 I=NPKL,NPKR
0060 2A0000   D     73          LHLD    NPKL
0063 5E             74          MOV     E,M
0064 23             75          INX     H
0065 56             76          MOV     D,M      ; D,E = NPKL
                    77 @3:
0066 2A0200   D     78          LHLD    NPKR
0069 EB             79          XCHG             ; (D,E) = NPKR
006A E5             80          PUSH    H        ; 1 H,L = I
006B C5             81          PUSH    B        ; 2 @FAC
006C CD0000   E     82          CALL    FQ0543   ; I > NPKR ?  (D,E)<H,L ?
006F C1             83          POP     B        ; 2
0070 E1             84          POP     H        ; 1
0071 DA9000   C     85          JC      @4       ;   YES
                    86 ; COMPARE COUNTS(I) WITH TEST
0074 E5             87          PUSH    .H       ; 1 I
0075 29             88          DAD     H
0076 29             89          DAD     H
0077 110C94         90          LXI     D,COUNTS-4
007A 19             91          DAD     D
007B EB             92          XCHG             ; (D,E) = COUNTS(I)
007C CD0000   E     93          CALL    FQFCM4   ; COMPARE @FAC:(D,E)
007F E620           94          ANI     20H      ; TEST < COUNTS(I) ?
0081 E1             95          POP     H        ; 1
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

0082 CA8B00   C    96             JZ      @5       ;   NO
0085 CD0000   E    97             CALL    FQFLD4   ;   YES, TEST=COUNTS(I)
0088 222400   D    98             SHLD    NPEAK    ;        NPEAK=I
                   99 @5:         ; 20 CONTINUE
008B 23           100             INX     H
008C EB           101             XCHG             ; D,E = I
008D C36600   C   102             JMP     @3
                  103 @4:
                  104 ; PRINT RESULTING MAX CHANNEL
0090 21ED02   C   105             LXI     H,MSG5
0093 CD0000   E   106             CALL    OUTSTR
0096 012400   D   107             LXI     B,NPEAK
0099 11C802   C   108             LXI     D,ZERO
009C CD0000   E   109             CALL    OUTINT
009F CD0000   E   110             CALL    CRLF
                  111 ; LEFT LIMIT = MIN0(20,NPEAK-2)
00A2 0614         112             MVI     B,20
00A4 2A2400   D   113             LHLD    NPEAK
00A7 7C           114             MOV     A,H
00A8 A7           115             ANA     A        ; NPEAK > 256 ?
00A9 C2B400   C   116             JNZ     LEFT     ;   YES
00AC 7D           117             MOV     A,L
00AD D602         118             SUI     2        ; A = NPEAK-2
00AF B8           119             CMP     B        ; ... < 20 ?
00B0 D2B400   C   120             JNC     LEFT     ;   NO
00B3 47           121             MOV     B,A      ;   YES, USE NPEAK-2
                  122 ; FIND LOCAL MINIMUM TO THE LEFT
                  123 ;   DO 100 I=1,LIMIT
                  124 ;     NLEFT=NPEAK+I
                  125 ;     IF(COUNTS(NLEFT).LE.COUNTS(NLEFT-1))
                  126 ;       GO TO 120
                  127 ;100  CONTINUE
                  128 LEFT:
00B4 78           129             MOV     A,B
00B5 2A2400   D   130             LHLD    NPEAK
00B8 E5           131             PUSH    H        ; 1 NPEAK
00B9 29           132             DAD     H
00BA 29           133             DAD     H
00BB 110894       134             LXI     D,COUNTS-8H
00BE 19           135             DAD     D
00BF EB           136             XCHG             ; (D,E)=COUNTS(NPEAK-1)
00C0 E1           137             POP     H        ; 1 (H,L) = NPEAK
                  138 NLOOP:
00C1 F5           139             PUSH    PSW      ; 1 A = LOOP COUNT
00C2 42           140             MOV     B,D
00C3 4B           141             MOV     C,E      ; (B,C)=COUNTS(NPEAK-I)
                  142             REPT    4
                  143             DCX     D
                  144             ENDM             ;(D,E)=COUNTS(NPEAK-I-1)
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

00C4 1B            145+         DCX     D
00C5 1B            146+         DCX     D
00C6 1B            147+         DCX     D
00C7 1B            148+         DCX     D
00C8 2B            149          DCX     H
00C9 222200    D   150          SHLD    NLEFT    ; NLEFT = NPEAK - I
00CC CD0000    E   151          CALL    FQ0684   ; COMPARE
00CF 07            152          RLC              ; (B,C) > (D,E) ?
00D0 D2DF00    C   153          JNC     @@120    ;   NO, THIS IS MINIMUM
00D3 F1            154          POP     PSW      ; 1
00D4 3D            155          DCR     A
00D5 C2C100    C   156          JNZ     NLOOP
00D8 21FC02    C   157          LXI     H,MSG6
00DB CD0000    E   158          CALL    OUTSTR
00DE F5            159          PUSH    PSW      ; 1
                   160 @@120:
00DF F1            161          POP     PSW      ; 1
00E0 C5            162          PUSH    B        ; 1 COUNTS(NLEFT)
                   163 ; RIGHT LIMIT = MIN0(10,NCHANL-NPEAK-1)
00E1 060A          164          MVI     B,10
00E3 2A0090        165          LHLD    NCHANL
00E6 EB            166          XCHG
00E7 2A2400    D   167          LHLD    NPEAK
00EA CD0000    E   168          CALL    FQ0588   ; H,L = NCHANL-NPEAK
00ED 2B            169          DCX     H
00EE 7C            170          MOV     A,H
00EF A7            171          ANA     A        ; H,L > 256 ?
00F0 C2F900    C   172          JNZ     RIGHT    ;   YES
00F3 7D            173          MOV     A,L
00F4 B8            174          CMP     B        ; H,L > 10 ?
00F5 D2F900    C   175          JNC     RIGHT    ;   YES
00F8 47            176          MOV     B,A
                   177 ; FIND MINIMUM TO RIGHT OF PEAK
                   178 ;   DO 130 I=1,LIMIT
                   179 ;      NRIGHT=NPEAK+I
                   180 ;      IF(COUNTS(NRIGHT).LE.COUNTS(NRIGHT+1))
                   181 ;         GO TO 140
                   182 ;130  CONTINUE
                   183 RIGHT:
00F9 78            184          MOV     A,B      ; A = LIMIT (LOOP COUNT)
00FA 2A2400    D   185          LHLD    NPEAK
00FD E5            186          PUSH    H        ; 2 NPEAK
00FE 29            187          DAD     H
00FF 29            188          DAD     H
0100 111094        189          LXI     D,COUNTS
0103 19            190          DAD     D
0104 EB            191          XCHG             ; (D,E)=COUNTS(NPEAK+1)
0105 E1            192          POP     H        ; 2
                   193 RLOOP:
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

0106 F5            194          PUSH    PSW       ; 2 LOOP COUNT
0107 42            195          MOV     B,D
0108 4B            196          MOV     C,E       ;(B,C)=COUNTS(NPEAK+I)
                   197          REPT    4
                   198          INX     D
                   199          ENDM              ;(D,E)=COUNTS(NPEAK+I+1)
0109 13            200+         INX     D
010A 13            201+         INX     D
010B 13            202+         INX     D
010C 13            203+         INX     D
010D 23            204          INX     H
010E 222600   D    205          SHLD    NRIGHT    ; NRIGHT = NPEAK + I
0111 CD0000   E    206          CALL    FQ0684    ; COMPARE
0114 07            207          RLC               ; (B,C) > (D,E) ?
0115 D22401   C    208          JNC     @@140     ;   NO, (B,C) IS MINIMUM
0118 F1            209          POP     PSW       ; 2
0119 3D            210          DCR     A
011A C20601   C    211          JNZ     RLOOP
011D 210A03   C    212          LXI     H,MSG7
0120 CD0000   E    213          CALL    OUTSTR
0123 F5            214          PUSH    PSW       ; 2
                   215 @@140:
0124 F1            216          POP     PSW       ; 2
                   217 ; SLOPE = (COUNTS(NRIGHT)-COUNTS(NLEFT))*1000
                   218 ; SLOPE = SLOPE/(NRIGHT-NLEFT)
0125 50            219          MOV     D,B
0126 59            220          MOV     E,C       ; (D,E) = COUNTS(NRIGHT)
0127 010600   D    221          LXI     B,@FAC
012A CD0000   E    222          CALL    FQFLD4
012D D1            223          POP     D         ; 1
012E CD0000   E    224          CALL    FQFSB4    ; FAC=CNT(NR)-CNT(NL)
0131 11C402   C    225          LXI     D,THOUS
0134 CD0000   E    226          CALL    FQFML4    ; FAC=FAC*1000
0137 2A2600   D    227          LHLD    NRIGHT
013A EB            228          XCHG
013B 2A2200   D    229          LHLD    NLEFT
013E CD0000   E    230          CALL    FQ0588    ; H,L = NRIGHT-NLEFT
0141 C5            231          PUSH    B         ; 1 ADDR OF FAC
0142 013600   D    232          LXI     B,TEMP
0145 CD0000   E    233          CALL    FQ0671    ; TEMP = FLOAT(H,L)
0148 50            234          MOV     D,B
0149 59            235          MOV     E,C
014A C1            236          POP     B         ; 1
014B CD0000   E    237          CALL    FQFDV4    ; FAC = FAC / TE
014E 112800   D    238          LXI     D,SLOPE
0151 CD0000   E    239          CALL    FQFST4    ; SLOPE = FAC
                   240 ; INTEGRATE UNDER PEAK
                   241 ;   PRINT INTEGRATION LIMITS (NLEFT,NRIGHT)
0154 211903   C    242          LXI     H,MSG8
```

```
LOC  OBJ          SEQ         SOURCE STATEMENT

0157 CD0000   E   243          CALL     OUTSTR
015A 11C802   C   244          LXI      D,ZERO
015D 012200   D   245          LXI      B,NLEFT
0160 CD0000   E   246          CALL     OUTINT
0163 0E2D         247          MVI      C,'-'
0165 CD0000   E   248          CALL     CO
0168 11C802   C   249          LXI      D,ZERO
016B 012600   D   250          LXI      B,NRIGHT
016E CD0000   E   251          CALL     OUTINT
0171 CD0000   E   252          CALL     CRLF
                  253 ;   INITIAL CONDITIONS
0174 2A0290       254          LHLD     NPEAKS
0177 29           255          DAD      H
0178 29           256          DAD      H
0179 E5           257          PUSH     H        ; 1
017A 11E890       258          LXI      D,TOTAL-4
017D 19           259          DAD      D
017E 223400   D   260          SHLD     @TOTAL   ; ADDR OF TOTAL(NPEAKS)
0181 010600   D   261          LXI      B,@FAC
0184 CD0000   E   262          CALL     FQFCLR   ; FAC = 0
0187 EB           263          XCHG
0188 CD0000   E   264          CALL     FQFST4   ; TOTAL = 0
018B E1           265          POP      H        ; 1
018C 11B091       266          LXI      D,AREA-4
018F 19           267          DAD      D
0190 223200   D   268          SHLD     @AREA    ; ADDR OF AREA(NPEAKS)
0193 EB           269          XCHG
0194 CD0000   E   270          CALL     FQFST4   ; AREA = 0
0197 111C00   D   271          LXI      D,FIRSTM
019A CD0000   E   272          CALL     FQFST4   ; FIRSTM = 0
                  273 ;   DO 200 I=NLEFT,NRIGHT
019D 2A2200   D   274          LHLD     NLEFT
01A0 222000   D   275          SHLD     I
01A3 29           276          DAD      H
01A4 29           277          DAD      H
01A5 110C94       278          LXI      D,COUNTS-4
01A8 19           279          DAD      D
01A9 EB           280          XCHG              ; (D,E) = COUNTS(NLEFT)
01AA 012C00   D   281          LXI      B,CNTNL  ; CNTNL = COUNTS(NLEFT)
01AD CD0000   E   282          CALL     FQFLD4
                  283 @14:
01B0 112600   D   284          LXI      D,NRIGHT
01B3 2A2000   D   285          LHLD     I
01B6 CD0000   E   286          CALL     FQ0543   ; I > NRIGHT ?
01B9 DA3302   C   287          JC       @15
                  288 ; TOTAL(NPEAKS) = TOTAL(NPEAKS) + COUNTS(I)
01BC 010600   D   289          LXI      B,@FAC
01BF 2A3400   D   290          LHLD     @TOTAL
01C2 E5           291          PUSH     H        ; 1 ADDR OF TOTAL
```

```
LOC  OBJ          SEQ         SOURCE STATEMENT

01C3 EB            292         XCHG
01C4 CD0000    E   293         CALL    FQFLD4
01C7 2A2000    D   294         LHLD    I
01CA 29            295         DAD     H
01CB 29            296         DAD     H
01CC 110C94        297         LXI     D,COUNTS-4
01CF 19            298         DAD     D
01D0 223000    D   299         SHLD    @COUNT  ; ADDR OF COUNTS(I)
01D3 EB            300         XCHG
01D4 CD0000    E   301         CALL    FQFAD4
01D7 D1            302         POP     D        ; 1
01D8 CD0000    E   303         CALL    FQFST4
                   304 ; ABOVE = COUNTS(I) - COUNTS(NLEFT)
                   305 ;         - SLOPE * (I-NLEFT) / 1000
01DB 2A2000    D   306         LHLD    I
01DE EB            307         XCHG
01DF 2A2200    D   308         LHLD    NLEFT
01E2 CD0000    E   309         CALL    FQ0588  ; H,L = I - NLEFT
01E5 CD0000    E   310         CALL    FQ0671  ; FAC = FLOAT(H,L)
01E8 112800    D   311         LXI     D,SLOPE
01EB CD0000    E   312         CALL    FQFML4  ; ... * SLOPE
01EE 11C402    C   313         LXI     D,THOUS
01F1 CD0000    E   314         CALL    FQFDV4  ; ... / 1000
01F4 CD0000    E   315         CALL    FQFNEG  ; ... -
01F7 2A3000    D   316         LHLD    @COUNT
01FA EB            317         XCHG
01FB CD0000    E   318         CALL    FQFAD4  ; + COUNT(I)
01FE 112C00    D   319         LXI     D,CNTNL
0201 CD0000    E   320         CALL    FQFSB4  ; - COUNT(NLEFT)
0204 111800    D   321         LXI     D,ABOVE
0207 CD0000    E   322         CALL    FQFST4  ; ABOVE = FAC
                   323 ; AREA(NPEAKS) = AREA(NPEAKS) + ABOVE
020A 2A3200    D   324         LHLD    @AREA
020D EB            325         XCHG
020E CD0000    E   326         CALL    FQFAD4
0211 CD0000    E   327         CALL    FQFST4
                   328 ; FIRSTM = FIRSTM + I*ABOVE
0214 2A2000    D   329         LHLD    I
0217 CD0000    E   330         CALL    FQ0671  ; FAC = FLOAT(I)
021A 111800    D   331         LXI     D,ABOVE
021D CD0000    E   332         CALL    FQFML4
0220 111C00    D   333         LXI     D,FIRSTM
0223 CD0000    E   334         CALL    FQFAD4
0226 CD0000    E   335         CALL    FQFST4
                   336 ; 200 CONTINUE (END OF DO LOOP)
0229 2A2000    D   337         LHLD    I
022C 23            338         INX     H
022D 222000    D   339         SHLD    I
0230 C3B001    C   340         JMP     @14
```

```
 LOC  OBJ          SEQ        SOURCE STATEMENT

                    341 @15:
                    342 ; REPORT RESULTS OF INTEGRATION (TOTAL,AREA)
0233 212903    C    343         LXI     H,MSG9
0236 CD0000    E    344         CALL    OUTSTR
0239 2A3400    D    345         LHLD    @TOTAL
023C 44             346         MOV     B,H
023D 4D             347         MOV     C,L
023E 11C802    C    348         LXI     D,ZERO
0241 CD0000    E    349         CALL    OUTFLT
0244 CD0000    E    350         CALL    CRLF
0247 213E03    C    351         LXI     H,MSG10
024A CD0000    E    352         CALL    OUTSTR
024D 2A3200    D    353         LHLD    @AREA
0250 44             354         MOV     B,H
0251 4D             355         MOV     C,L
0252 11C802    C    356         LXI     D,ZERO
0255 CD0000    E    357         CALL    OUTFLT
                    358 ;
0258 2A0290         359         LHLD    NPEAKS
025B 29             360         DAD     H
025C 29             361         DAD     H
025D 112090         362         LXI     D,CHANL-4
0260 19             363         DAD     D
0261 E5             364         PUSH    H        ; 1 CHANL(NPEAKS)
                    365 ; IF(AREA(NPEAKS).LE.0) THEN
0262 010600    D    366         LXI     B,@FAC
0265 2A3200    D    367         LHLD    @AREA
0268 EB             368         XCHG
0269 CD0000    E    369         CALL    FQFLD4
026C 11C802    C    370         LXI     D,ZERO
026F CD0000    E    371         CALL    FQ0684   ; COMPARE
0272 07             372         RLC              ; AREA(NPEAKS) > 0 ?
0273 DA9202    C    373         JC      @16      ;   YES
                    374 ;   PRINT "*CAUTION*"
0276 214F03    C    375         LXI     H,MSG11
0279 CD0000    E    376         CALL    OUTSTR
                    377 ;   CHANL(NPEAKS) = NPEAK * 10
027C 2A2400    D    378         LHLD    NPEAK
027F 010600    D    379         LXI     B,@FAC
0282 CD0000    E    380         CALL    FQ0671   ; FAC = FLOAT(NPEAK)
0285 11C002    C    381         LXI     D,TEN
0288 CD0000    E    382         CALL    FQFML4
028B D1             383         POP     D        ; 1
028C CD0000    E    384         CALL    FQFST4   ; CHANL(NPEAKS) = FAC
028F C3BA02    C    385         JMP     @17
                    386 ; ELSE
                    387 @16:
                    388 ;   CHANL(NPEAKS) = 10 * FIRSTM/AREA(NPEAKS)
0292 11C002    C    389         LXI     D,TEN
```

```
LOC  OBJ          SEQ        SOURCE STATEMENT

0295 CD0000    E   390           CALL    FQFLD4
0298 111C00    D   391           LXI     D,FIRSTM
029B CD0000    E   392           CALL    FQFML4
029E 2A3200    D   393           LHLD    @AREA
02A1 EB            394           XCHG
02A2 CD0000    E   395           CALL    FQFDV4
02A5 D1            396           POP     D        ; 1
02A6 D5            397           PUSH    D        ; 1 CHANL(NPEAKS)
02A7 CD0000    E   398           CALL    FQFST4
                   399 ;    PRINT RESULT (CHANL(NPEAKS) )
02AA CD0000    E   400           CALL    CRLF
02AD 215C03    C   401           LXI     H,MSG12
02B0 CD0000    E   402           CALL    OUTSTR
02B3 C1            403           POP     B        ; 1
02B4 11BE02    C   404           LXI     D,HUNDRD
02B7 CD0000    E   405           CALL    OUTFLT
                   406 ; ENDIF
                   407 @17:
02BA CD0000    E   408           CALL    CRLF
02BD C9            409           RET
                   410 ;
                   411 ;****************************************
                   412 ;
                   413 ; CONSTANTS
                   414 ;
02BE 6400          415 HUNDRD: DW      100
02C0 0A00          416 TEN:    DW      10,0
02C2 0000
02C4 E803          417 THOUS:  DW      1000,0
02C6 0000
02C8 0000          418 ZERO:   DW      0,0
02CA 0000
                   419 ;
                   420 ; VARIABLES
                   421 ;
                   422           DSEG
                   423 ; PARAMETER ADDRESSES:  ORDER DEPENDENT
0002               424 NPKL:   DS      2
0002               425 NPKR:   DS      2
0002               426 LASTR:  DS      2
                   427 ;
                   428 ; LOCAL VARIABLES
0012               429 @FAC:   DS      18
0004               430 ABOVE:  DS      4
0004               431 FIRSTM: DS      4
0002               432 I:      DS      2
0002               433 NLEFT:  DS      2
0002               434 NPEAK:  DS      2
0002               435 NRIGHT: DS      2
```

```
LOC  OBJ          SEQ         SOURCE STATEMENT

0004               436 SLOPE:  DS      4
                   437 ;
0004               438 CNTNL:  DS      4
0002               439 @COUNT: DS      2
0002               440 @AREA:  DS      2
0002               441 @TOTAL: DS      2
0004               442 TEMP:   DS      4
                   443 ;
                   444 ; CHARACTER CONSTANTS
                   445 ;
                   446         CSEG
02CC 5045414B      447 MSG1:   DB      'PEAK #',0
02D0 2023
02D2 00
02D3 20464F55      448 MSG2:   DB      ' FOUND ',0
02D7 4E4420
02DA 00
02DB 2A4C494D      449 MSG4:   DB      '*LIMIT* LAST PEAK',0
02DF 49542A20
02E3 4C415354
02E7 20504541
02EB 4B
02EC 00
02ED 2020204D      450 MSG5:   DB      '   MAXIMUM IN ',0
02F1 4158494D
02F5 554D2049
02F9 4E20
02FB 00
02FC 2020204C      451 MSG6:   DB      '   LEFT LIMIT',0
0300 45465420
0304 4C494D49
0308 54
0309 00
030A 20202052      452 MSG7:   DB      '   RIGHT LIMIT',0
030E 49474854
0312 204C494D
0316 4954
0318 00
0319 20202049      453 MSG8:   DB      '   INTEGRATING ',0
031D 4E544547
0321 52415449
0325 4E4720
0328 00
0329 20202020      454 MSG9:   DB      '    TOTAL COUNTS = ',0
032D 20544F54
0331 414C2043
0335 4F554E54
0339 53203D20
033D 00
```

```
 LOC  OBJ          SEQ         SOURCE STATEMENT

 033E 20202020      455 MSG10:  DB       '     NET AREA = ',0
 0342 204E4554
 0346 20415245
 034A 41203D20
 034E 00
 034F 2020202A      456 MSG11:  DB       '   *CAUTION*',0
 0353 43415554
 0357 494F4E2A
 035B 00
 035C 20202050      457 MSG12:  DB       '   PEAK AT CHANNEL ',0
 0360 45414B20
 0364 41542043
 0368 48414E4E
 036C 454C20
 036F 00
                    458 ;
                    459 ; BLANK COMMON
                    460         ASEG
 9000               461         ORG     9000H
 0002               462 NCHANL: DS      2
 0002               463 NPEAKS: DS      2
 0020               464 COEFF:  DS      8*4
 00C8               465 CHANL:  DS      50*4
 00C8               466 TOTAL:  DS      50*4
 00C8               467 AREA:   DS      50*4
 00C8               468 PKERGY: DS      50*4
 00C8               469 PGPS:   DS      50*4
 0004               470 TIME:   DS      4
 3FFC               471 COUNTS: DS      4095*4
                    472 ;
                    473         END

PUBLIC SYMBOLS
PKPROC C 0000

EXTERNAL SYMBOLS
CO     E 0000   CRLF   E 0000   FQ0543 E 0000   FQ0588 E 0000
FQ0664 E 0000   FQ0671 E 0000   FQ0684 E 0000   FQ0686 E 0000
FQFAD4 E 0000   FQFCLR E 0000   FQFCM4 E 0000   FQFDV4 E 0000
FQFLD4 E 0000   FQFML4 E 0000   FQFNEG E 0000   FQFSB4 E 0000
FQFST4 E 0000   OUTFLT E 0000   OUTINT E 0000   OUTSTR E 0000

USER SYMBOLS
@1     C 005A   @14    C 01B0   @15    C 0233   @16    C 0292
@17    C 02BA   @3     C 0066   @4     C 0090   @5     C 008E
@@120  C 00DF   @@140  C 0124   @AREA  D 0032   @COUNT D 0030
@FAC   D 0006   @TOTAL D 0034   ABOVE  D 0018   AREA   A 91B4
CHANL  A 9024   CNTNL  D 002C   CO     E 0000   COEFF  A 9004
COUNTS A 9410   CRLF   E 0000   FIRSTM D 001C   FQ0543 E 0000
```

FQ0588	E	0000	FQ0664	E	0000	FQ0671	E	0000	FQ0684	E	0000
FQ0686	E	0000	FQFAD4	E	0000	FQFCLR	E	0000	FQFCM4	E	0000
FQFDV4	E	0000	FQFLD4	E	0000	FQFML4	E	0000	FQFNEG	E	0000
FQFSB4	E	0000	FQFST4	E	0000	HUNDRD	C	02BE	I	D	0020
LASTR	D	0004	LEFT	C	00B4	MSG1	C	02CC	MSG10	C	033E
MSG11	C	034F	MSG12	C	035C	MSG2	C	02D3	MSG4	C	02DB
MSG5	C	02ED	MSG6	C	02FC	MSG7	C	030A	MSG8	C	0319
MSG9	C	0329	NCHANL	A	9000	NLEFT	D	0022	NLOOP	C	00C1
NPEAK	D	0024	NPEAKS	A	9002	NPKL	D	0000	NPKR	D	0002
NRIGHT	D	0026	OUTFLT	E	0000	OUTINT	E	0000	OUTSTR	E	0000
PGPS	A	9344	PKERGY	A	927C	PKPROC	C	0000	RIGHT	C	00F9
RLOOP	C	0106	SLOPE	D	0028	TEMP	D	0036	TEN	C	02C0
THOUS	C	02C4	TIME	A	940C	TOTAL	A	90EC	ZERO	C	02C8

ASSEMBLY COMPLETE, NO ERRORS

ISIS-II FORTRAN-80 V1.0 COMPILATION OF PROGRAM UNIT PKSRCH
OBJECT MODULE PLACED IN PKSRCH.OBJ
COMPILER INVOKED BY: :F1:FORT80 PKSRCH.FOR PAGELENGTH(54)

```
$PAGEWIDTH(79) DATE(13 JAN 81)
$TITLE('PKSRCH - SCAN SPECTRUM FOR PEAKS')
      SUBROUTINE PKSRCH
C SEARCH FOR PEAKS IN THE RANGE:  COUNTS(4) TO COUNTS(NCHANL-3)
C   MULTIPLY COUNTS BY ZERO-AREA CORRELATOR TO MEASURE CURVATURE
C   COMPUTE TEST=CURV*SQRT(50 CHANNEL FORWARD AVERAGE)
C   NLEFT = CHANNEL WHERE CURVATURE FIRST EXCEEDS TEST
C   NRIGHT = CHANNEL WHERE CURVATURE LAST EXCEEDS TEST
C
C NOTE:  FWHM EXPECTED BY THIS ROUTINE IS 3 CHANNELS
C
      COMMON NCHANL,NPEAKS,COEFF(8)
      COMMON CHANL(50),TOTAL(50),AREA(50),PKERGY(50),PGPS(50)
      COMMON TIME,COUNTS(4095)
      IF(NCHANL.EQ.0) RETURN
C ASK USER FOR CURVATURE CRITERIA
10    CALL OUTCHR('PEAK DETECT CURVATURE? (1-1000)>')
      CALL ININT(ICURV,12)
      IF(ICURV.LT.1.OR.ICURV.GT.1000) GO TO 10
      CURV=ICURV*ICURV
C
      NLEFT=1
      LASTR=0
      SUM=0
      NPEAKS=0
      DO 30 I=1,50
30      SUM=SUM+COUNTS(I)
      T=CURV*SUM/50
      DO 140 M=6,NCHANL-4
        C=0
        DO 100 I=1,3
          J=M+I
          C=C-COUNTS(J-6)+COUNTS(J-3)+COUNTS(J-3)-COUNTS(J)
100       CONTINUE
        C=C*ABS(C)
        IF(M.GT.6) THEN
          IF(M.LE.NCHANL-53) THEN
            SUM=SUM-COUNTS(M-6)+COUNTS(M+44)
            T=CURV*SUM/50
          ENDIF
C TEST FOR PEAK BOUNDARY
          IF(C.GT.T) THEN
            IF(C1.LE.T1) NLEFT=M
          ELSE IF(C1.GT.T1) THEN
            NRIGHT=M
```

```
                  CALL PKPROC(NLEFT,NRIGHT,LASTR)
                  NLEFT=0
                ENDIF
              ENDIF
              C1=C
              T1=T
  140         CONTINUE
            RETURN
            END
```

MODULE INFORMATION:

```
     CODE AREA SIZE     = 0299H    665D
     VARIABLE AREA SIZE = 003AH     58D
     MAXIMUM STACK SIZE = 0006H      6D
     54 LINES READ

     0 PROGRAM ERROR(S) IN PROGRAM UNIT PKSRCH

     0 TOTAL PROGRAM ERROR(S)
     END OF FORTRAN COMPILATION
```

Appendix B

Interface Testing

The design and construction of the demonstration system interface board was described in Chapter 2 of the body of this report. This appendix describes the procedures, programs, and results of testing the interface board.

A program called PPTEST (for Parallel Port TEST) was written which

- initializes the microcomputer parallel port hardware,
- asks the user which input port to test,
- asks the user what value to put on the output port,
- puts the value specified on the output lines,
- prints the value of the input port specified, and
- alternately displays the value of each half of the input port on a one digit hexadecimal light emitting diode (LED) display.

An assembly listing of the program follows this discussion.

The program was used to bench test the interface. With the program running, the output pins of the jack J5 were checked with a logic probe. The output driver in the single board computer inverts the value sent to it; so, when a hexadecimal FF is sent out, all output lines should have a low logic value. Then, a hexadecimal 00 is sent and all lines checked for high value.

The input lines were checked by sending a hexadecimal

01 to the output port to select the jack J4 input lines. For each input port, each input line was grounded in turn. The pins normally float to +5 volts which, after inversion, represents a 0 in that bit position of the hexadecimal display. Grounding the pin corresponding to bit 0 of the port caused the display to change from 0 followed by 0 to 0 followed by 1. The two numbers flash alternately on the display. The first number can be identified because it is displayed for 3 tenths of a second, while the second number is displayed for 9 tenths of a second. Grounding the jack J4 pin corresponding to bit 1 of the input port results in a 0 followed by 2 on the display. Other pins are checked in the same manner until the bit 7 pin produces an 8 followed by 0 on the display. The bench testing revealed a few wiring errors, some bad components, and revealed that all outputs and some inputs are inverted by the single board computer.

The next phase of testing involved writing a short program named CHIN (for CHannels INto microcomputer) to transfer multichannel analyzer data over the interface. No conversion of the binary coded decimal data is attempted at this point. A program listing is included at the end of this appendix, but should be used with caution because this program was abandoned before all revisions were incorporated. While the program ran, the values of the RDY2 and CMD lines of the multichannel analyzer were monitored with an oscilloscope. Through modifications to the CHIN program and observation of the oscilloscope trace, the actual response of the multichannel analyzer was determined. The results are reported in chapter 2 of the body of this report.

The actual response of the analyzer caused a change in the program which runs the interface. In particular, the response of the analyzer was found to be faster than the microcomputer, and so there was no need to have the analyzer interupt the microcomputer when channel data was ready.

ASM :F1:PPTEST.SRC PAGELENGTH(60) PAGEWIDTH(79)


```
 LOC  OBJ          SEQ          SOURCE STATEMENT

                     1 $MACROFILE
                     2 $TITLE('PPTEST - TEST THE PARALLEL PORT INTERFACE')
                     3 ;
                     4 ; 19 DEC 80
                     5 ;
                     6 ; 1. INPUT PORT 1 AND PRINT VALUE
                     7 ; 2. GET OUTPUT VALUE FROM TELETYPE
                     8 ; 3. OUTPUT VALUE TO PORT 3
                     9 ;
                    10 ; THIS ROUTINE NOT DESIGNED TO RUN ON DEVELOPMENT SYSTE
                       M
                    11 ;
                    12         NAME    PPTEST
                    13 ;
                    14 ; EQUATES TO MONITOR ROUTINES
                    15 ;
 0487               16 NMOUT   EQU     0487H
 0312               17 CROUT   EQU     0312H
 034F               18 GETHX   EQU     034FH
 0318               19 DELAY   EQU     0318H
 0307               20 CO      EQU     0307H
                    21 ;
                    22 ; OTHER EQUATES
                    23 ;
 00E4               24 PORTIN  EQU     0E4H
 00E6               25 PRTOUT  EQU     0E6H
 00E7               26 PORTC1  EQU     0E7H
 00EB               27 PORTC2  EQU     0EBH
 0090               28 PCW123  EQU     90H      ; 10010010 = A:MODE 0-I B:MODE
                       0-I C:MODE: 0-0
 009B               29 PCW456  EQU     9BH      ; 10011011 = A:MODE 0-I B:MODE
                       0-I C:MODE: 0-I
                    30 ;
                    31 ;****************************************************
                       *****************
                    32 ;
 8000               33         ORG     8000H    ; MOVE TO RAM TO EXECUTE
                    34 ;
                    35 INIT:                    ; INITIALIZE PARALLEL PORTS
 8000 3E90          36         MVI     A,PCW123
 8002 D3E7          37         OUT     PORTC1
 8004 3E9B          38         MVI     A,PCW456
 8006 D3EB          39         OUT     PORTC2
                    40 ;
                    41 INPUT:
 8008 214880        42         LXI     H,INMSG
```

```
LOC  OBJ         SEQ        SOURCE STATEMENT

800B CD2E80       43          CALL    OUTSTR
800E 3E01         44          MVI     A,1
8010 CD3980       45          CALL    DELAY1
8013 DBE4         46          IN      PORTIN
8015 CD8704       47          CALL    NMOUT
8018 CD1203       48          CALL    CROUT
                  49 ;
                  50 OUTPUT:
801B 214F80       51          LXI     H,OUTMSG
801E CD2E80       52          CALL    OUTSTR
8021 CD4F03       53          CALL    GETHX
8024 D3E6         54          OUT     PRTOUT
8026 3E05         55          MVI     A,5
8028 CD3980       56          CALL    DELAY1
802B C30880       57          JMP     INPUT
                  58 ;
                  59 ;*****************************************************
                     *******************
                  60 ; NAME: OUTSTR
                  61 ; INPUTS:         H,L = ADDRESS OF STRING TO OUTPUT
                  62 ; OUTPUTS:        NONE
                  63 ; CALLS:          CO
                  64 ; DESTROYS:       A,C,H,L
                  65 ; FUNCTION:       OUTPUT STRING TERMINATED BY ZERO BYTE T
                     O CONSOLE
                  66 ;
                  67 OUTSTR:
802E AF           68          XRA     A       ; A = 0
802F 4E           69          MOV     C,M     ; C = NEXT CHAR
8030 B9           70          CMP     C       ;   = 0?
8031 C8           71          RZ              ;     YES, RETURN
8032 CD0703       72          CALL    CO      ; OUTPUT CHAR TO CONSOLE
8035 23           73          INX     H       ; POINT TO NEXT CHAR
8036 C32E80       74          JMP     OUTSTR
                  75 ;
                  76 ;*****************************************************
                     ********************
                  77 ;
                  78 ;NAME:  DELAY1
                  79 ;INPUTS:          A = NUMBER OF CHARACTERS
                  80 ; OUTPUTS:        NONE
                  81 ; CALLS:          DELAY
                  82 ; DESTROYS:       A,B,C
                  83 ; FUNCTION:       DELAY AN INTEGRAL NUMBER OF SECONDS
                  84 ;
                  85 DELAY1:
8039 01E803       86          LXI     B,1000
                  87 DELOOP:
803C CD1803       88          CALL    DELAY   ; DELAY 1 MILLISECOND
```

```
 LOC  OBJ          SEQ         SOURCE STATEMENT

 803F 0B             89          DCX     B        ;  ... 1000 TIMES
 8040 C23C80         90          JNZ     DELOOP
 8043 3D             91          DCR     A        ; SECONDS LEFT ?
 8044 C23980         92          JNZ     DELAY1   ;   YES, LOOP
 8047 C9             93          RET
                     94 ;
                     95 ;*****************************************************
                       *********************
                     96 ;
                     97 ; STORAGE AND CONSTANTS
                     98 ;
 8048 494E5055       99 INMSG:  DB      'INPUT=',0
 804C 543D
 804E 00
 804F 4F555450      100 OUTMSG: DB      'OUTPUT=?',0
 8053 55543D3F
 8057 00
 8000               101          END     INIT

PUBLIC SYMBOLS


EXTERNAL SYMBOLS


USER SYMBOLS
CO     A 0307    CROUT  A 0312    DELAY  A 0318    DELAY1 A 8039
DELOOP A 803C    GETHX  A 034F    INIT   A 8000    INMSG  A 8048
INPUT  A 8008    NMOUT  A 0487    OUTMSG A 804F    OUTPUT A 801B
OUTSTR A 802E    PCW123 A 0090    PCW456 A 009B    PORTC1 A 00E7
PORTC2 A 00EB    PORTIN A 00E4    PRTOUT A 00E6

ASSEMBLY COMPLETE,  NO ERRORS
```

```
ASM :F1:CHIN.SRC PAGELENGTH(60) PAGEWIDTH(79)


ISIS-II 8080/8085 MACRO ASSEMBLER, V2.0        MODULE   PAGE    1
CHIN - INPUT CHANNEL CONTENTS FROM MULTI-CHANNEL ANALYZER

 LOC  OBJ         SEQ          SOURCE STATEMENT

                    1 $MACROFILE
                    2 $TITLE('CHIN - INPUT CHANNEL CONTENTS FROM MULTI-CHANNE
                      L ANALYZER')
                    3 ;
                    4 ; 6 OCT 80
                    5 ;
                    6 ; 1. INPUT CHANNELS 0-2047 FROM MCA
                    7 ; 2. STORE BCD IN RAM LOCATIONS A000-BFFF
                    8 ;    (4 BYTES/CHANNEL)
                    9 ; 3. IDENTIFY EVERY TENTH CHANNEL IN LED DIGIT
                   10 ;
                   11 ; THIS ROUTINE NOT DESIGNED TO RUN  ON
                   12 ; DEVELOPMENT SYSTEM
                 = 13 $INCLUDE(MONADD.EQU)
                 = 14 ;
                 = 15 ; EQUATES TO USEFUL SBC MONITOR ROUTINES
                 = 16 ;
 02AF            = 17 ADRD     EQU      02AFH    ; CONSOLE <-- ADDRESS IN H,L RE
                      GISTERS
 02D9            = 18 BYTE     EQU      02D9H    ; READ 2 ASCII HEX DIGITS IN A
 02F4            = 19 CI       EQU      02F4H    ; A <-- CONSOLE CHARACTER
 02FE            = 20 CNVBN    EQU      02FEH    ; A <-- BINARY VALUE OF HEX IN
                      C
 0307            = 21 CO       EQU      0307H    ; CONSOLE <-- ASCII CHAR IN C
 0312            = 22 CROUT    EQU      0312H    ; CONSOLE <-- CR LF
 0318            = 23 DELAY    EQU      0318H    ; 1 MILLISECOND
 0321            = 24 ECHO     EQU      0321H    ; CONSOLE <-- ASCII CHAR IN C C
                      R->CRLF ESC->$
 0348            = 25 GETCH    EQU      0348H    ; C <-- CONSOLE 1 CHAR W/O PARI
                      TY
 034F            = 26 GETHX    EQU      034FH    ; BC <-- 4 DIGIT HEX D=DELIM CA
                      RRY=1 NONULL
 03C8            = 27 HILO     EQU      03C8H    ; CARRY<-1 IF HL>=DE UNSIGNED
 0487            = 28 NMOUT    EQU      0487H    ; CONSOLE <-- 2 HEX DIGITS FROM
                       C
 050D            = 29 PRVAL    EQU      050DH    ; A <-- ASCII HEX DIGIT IN LOW
                      ORDER A
 0548            = 30 REGSV    EQU      0548H    ; SAVE REGISTERS ON INTERUPT
 05B3            = 31 RSTTF    EQU      05B3H    ; RESTORE REGISTERS
 0609            = 32 VALDG    EQU      0609H    ; CARRY <- 1 IF C VALID HEX DIG
                      IT
 0624            = 33 VALDL    EQU      0624H    ; CARRY <- 1 IF C  = , CR BLANK
                 = 34 ;
                   35 ;
                   36 ; I/O EQUATES
                   37 ;
```

```
LOC  OBJ        SEQ         SOURCE STATEMENT

00EA              38 IN2     EQU    0EAH      ; 100K,10K DIGITS
00E9              39 IN1     EQU    0E9H      ; 1000,100 DIGITS
00E8              40 IN0     EQU    0E8H      ; 10,1 DIGITS
00E6              41 OUTBYT  EQU    0E6H      ; EXT-R,EXT-A,RDY-2,SM,SELECT
00E5              42 DISPL   EQU    0E5H      ; LED DISPLAY
00E7              43 OUTCTL  EQU    0E7H      ; PORT 1,2,3 CONTROL
00E7              44 OUTBIT  EQU    0E7H      ; OUTPUT 1 BIT
00EB              45 INCTL   EQU    0EBH      ; PORT 4,5,6 CONTROL
00D6              46 LED     EQU    0D6H      ; DIAGNOSTIC LED ON SBC BOARD
0090              47 OUTCW   EQU    10010000B        ; MODE=0 1-1 2-0 3-0
009B              48 INCW    EQU    10011011B        ; MODE=0 INPUT
0085              49 RLOW    EQU    10000101B        ; READY2 LOW = SET BIT
                     2
0084              50 RHIGH   EQU    10000100B        ; READY2 HIGH = RESET B
                     IT 2
0080              51 ADDRIN  EQU    10000000B        ; INPUT ADDRESS & STATU
                     S
0081              52 DATAIN  EQU    10000001B        ; INPUT DATA
                  53 ;
                  54 ; INTERUPT EQUATES
                  55 ;
0061              56 EOI1    EQU    01100001B        ; RESET INTERUPT 1
00D8              57 INTC    EQU    0D8H             ; INTERUPT COMMAND PORT
3FE1              58 INTABL  EQU    3FE1H            ; ADDRESS OF INTERUPT T
                     ABLE
                  59 ;
                  60 ;******************************************
                  61 ;
8000              62         ORG    8000H            ; MOVE TO RAM TO EXECUT
                     E
                  63 ;
                  64 INIT:
8000 3E90         65         MVI    A,OUTCW
8002 D3E7         66         OUT    OUTCTL           ; INITIALIZE PORTS 1,2,
                     3
8004 3E9B         67         MVI    A,INCW
8006 D3EB         68         OUT    INCTL            ; INITIALIZE PORTS 4,5,
                     6
8008 3E12         69         MVI    A,10010B         ; SCANMASTER OFF, READO
                     UT
800A D3E6         70         OUT    OUTBYT
800C 213680       71         LXI    H,HANDL1
800F 22E53F       72         SHLD   INTABL+1*4       ; INTERUPT 1 TO MY HAND
                     LER
                  73 ;
8012 2100A0       74         LXI    H,0A000H         ; ADDRESS OF FIRST CHAN
                     NEL
                  75 CHANL:                          ; INPUT NEXT CHANNEL
8015 3E85         76         MVI    A,RLOW
```

```
 LOC  OBJ         SEQ        SOURCE STATEMENT

 8017 D3E7         77         OUT     OUTBIT          ; MAKE READY2 LOW
 8019 76           78         HLT                     ; WAIT FOR INTERUPT
 801A CD4D80       79         CALL    INPCH           ; INPUT THE CHANNEL
 801D CD6180       80         CALL    IDENT           ; DISPLAY ADDRESS IF I
                      LOW
 8020 7C           81         MOV     A,H
 8021 FEC0         82         CPI     0C0H            ; H,L = C000 ?
 8023 CA3280       83         JZ      DONE            ;   YES, DONE
                   84 ; DELAY 50 MS. FOR DEBUG PURPOSES
 8026 3E32         85         MVI     A,50
                   86 DELOOP:
 8028 CD1803       87         CALL    DELAY
 802B 3D           88         DCR     A
 802C C22880       89         JNZ     DELOOP
 802F C31580       90         JMP     CHANL           ; INPUT NEXT CHANNEL
                   91 DONE:                           ; LAST CHANNEL IS IN RA
                      M
 8032 AF           92         XRA     A
 8033 D3E6         93         OUT     OUTBYT          ; SCANMASTER ON, STOP M
                      ODE
 8035 CF           94         RST     1               ; EXIT TO MONITOR
                   95 ;
                   96 ;********************************************
                   97 ;
                   98 HANDL1:                         ; INTERUPT 1 HANDLER
                   99 ;
                  100 ; INPUTS:       NONE
                  101 ; OUTPUTS:      NONE
                  102 ; CALLS:        NONE
                  103 ; DESTROYS:     NONE
                  104 ;
 8036 F5          105         PUSH    PSW             ; PRESERVE A,FLAGS
 8037 3E84        106         MVI     A,RHIGH
 8039 D3E7        107         OUT     OUTBIT          ; READY2 <-- HIGH
 803B 3E80        108         MVI     A,ADDRIN        ; SELECT ADDRESS SIDE O
                      F MUX
 803D D3E7        109         OUT     OUTBIT
                  110 CMDTST:
 803F DBEA        111         IN      IN2
 8041 E602        112         ANI     10B             ; COMMAND STILL HIGH ?
 8043 CA3F80      113         JZ      CMDTST          ;   YES, WAIT FOR LOW
 8046 3E61        114         MVI     A,EOI1
 8048 D3D8        115         OUT     INTC            ; RESET INTERUPT
 804A F1          116         POP     PSW             ; RESTORE A,FLAGS
 804B FB          117         EI
 804C C9          118         RET
                  119 ;
                  120 ;********************************************
                  121 ;
```

```
LOC  OBJ         SEQ        SOURCE STATEMENT

                 122 INPCH:                              ; INPUT ONE CHANNEL TO
                     MEMORY
                 123 ;
                 124 ; INPUTS:       H,L = ADDRESS 4 BYTE BLOCK
                 125 ; OUTPUTS:      H,L = ADDRESS NEXT BLOCK
                 126 ; CALLS:        NONE
                 127 ; DESTROYS:     A
                 128 ;
804D 3E81        129         MVI     A,DATAIN
804F D3E7        130         OUT     OUTBIT              ; SELECT DATA SIDE OF M
                     UX
8051 AF          131         XRA     A
8052 77          132         MOV     M,A                 ; FIRST BYTE = 0
8053 DBEA        133         IN      IN2
8055 23          134         INX     H
8056 77          135         MOV     M,A                 ; SECOND BYTE = 100K 10
                     K
8057 DBE9        136         IN      IN1
8059 23          137         INX     H
805A 77          138         MOV     M,A                 ; THIRD BYTE = 1000 100
805B DBE8        139         IN      IN0
805D 23          140         INX     H
805E 77          141         MOV     M,A                 ; FOURTH BYTE = 10 1
805F 23          142         INX     H                   ; POINT TO FIRST BYTE O
                     F NEXT BLOCK
8060 C9          143         RET
                 144 ;
                 145 ;*********************************************
                 146 ;
                 147 IDENT:                              ; IF I=LOW DISPLAY ADDR
                     ESS DIGIT
                 148 ;
                 149 ; INPUTS:       NONE
                 150 ; OUTPUTS:      NONE
                 151 ; CALLS:        NONE
                 152 ; DESTROYS:     NONE
                 153 ;
8061 D3D6        154         OUT     LED                 ; FLASH LED TO SAY WE'R
                     E ALIVE
                 155 ; CHECK IDENTIFY LINE
8063 3E80        156         MVI     A,ADDRIN
8065 D3E7        157         OUT     OUTBIT              ; SELECT ADDRESS SIDE O
                     F MUX
8067 DBE9        158         IN      IN1
8069 E680        159         ANI     10000000B           ; I LOW ?
806B C0          160         RNZ                         ;   NO, DON'T IDENTIFY
                 161 ; DISPLAY DIGIT
806C DBE8        162         IN      IN0                 ; READ A10,A1 DIGITS
                 163         REPT    4
```

```
 LOC  OBJ         SEQ         SOURCE STATEMENT

                   164         RRC
                   165         ENDM
 806E 0F           166+        RRC
 806F 0F           167+        RRC
 8070 0F           168+        RRC
 8071 0F           169+        RRC
 8072 D3E5         170         OUT     DISPL           ; OUTPUT THE DIGIT TO H
                       EX LED
 8074 C9           171         RET
                   172 ;
 8000              173         END     INIT

PUBLIC SYMBOLS


EXTERNAL SYMBOLS


USER SYMBOLS
ADDRIN A 0080   ADRD   A 02AF   BYTE   A 02D9   CHANL  A 8015
CI     A 02F4   CMDTST A 803F   CNVBN  A 02FE   CO     A 0307
CROUT  A 0312   DATAIN A 0081   DELAY  A 0318   DELOOP A 8028
DISPL  A 00E5   DONE   A 8032   ECHO   A 0321   EOI1   A 0061
GETCH  A 0348   GETHX  A 034F   HANDL1 A 8036   HILO   A 03C8
IDENT  A 8061   IN0    A 00E8   IN1    A 00E9   IN2    A 00EA
INCTL  A 00EB   INCW   A 009B   INIT   A 800C   INPCH  A 804D
INTABL A 3FE1   INTC   A 00D8   LED    A 00D6   NMOUT  A 0487
OUTBIT A 00E7   OUTBYT A 00E6   OUTCTL A 00E7   OUTCW  A 0090
PRVAL  A 050D   REGSV  A 0548   RHIGH  A 0084   RLOW   A 0085
RSTTF  A 05B3   VALDG  A 0609   VALDL  A 0624

ASSEMBLY COMPLETE,  NO ERRORS
```

Vita

DeFrance Clarke III was born 4 June 1948 in Hartford, Connecticut to DeFrance Clarke, Jr. and Elizabeth Jane Wright Clarke. He graduated from Blair Academy in 1966 and received a Bachelor of Science degree in Physics, cum laude, from Yale University in 1970. A Master of Science degree in Computer Science was conferred upon him by the University of Arizona in 1975. Entering the United States Air Force in 1970, he was trained first as an officer, then as a Navigator and Electronic Warfare Officer. He controlled Remotely Piloted Vehicles from 1972 until 1977 and then taught Electronic Warfare and trained programmers until 1979 when he was selected for the Graduate Nuclear Engineering program in the Engineering School of the Air Force Institute of Technology. He is a member of the Association of Old Crows, the American Nuclear Society, and Tau Beta Pi.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/GNE/PH/81M-2	2. GOVT ACCESSION NO. AD-A106	3. RECIPIENT'S CATALOG NUMBER 176
4. TITLE (and Subtitle) FEASIBILITY OF INTERFACING A MICROCOMPUTER WITH A MULTICHANNEL ANALYZER TO PERFORM GAMMA RAY SPECTROSCOPY		5. TYPE OF REPORT & PERIOD COVERED
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) DeFrance Clarke III, Captain, USAF		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology (AFIT-EN) Wright-Patterson AFB, OH 45433		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE March 1981
		13. NUMBER OF PAGES 138
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

APPROVED FOR PUBLIC RELEASE AFR 190-17.

18. SUPPLEMENTARY NOTES

20 OCT 1981

Frederick C. Lynch, Major, USAF
Director of Public Affairs

Fredric C. Lynch

FREDRIC C. LYNCH, Major, USAF
Director of Public Affairs

Air Force Institute of Technology (ATC)
Wright-Patterson AFB, OH 45433

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Gamma Ray Spectroscopy
Pulse Height Analyzers
Microcomputers
Interfaces
Arithmetic
Integer Programming

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Interfacing a microcomputer to a Nuclear Data Series 2200 multichannel analyzer has been demonstrated. Programs for data transfer, peak detection, peak net area integration, peak centroid location, and energy calibration

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Block 20

were written. Data transfer and peak detection scan were demonstrated on 4096 channel spectra in less than 5 minutes. Multibyte integer arithmetic routines for the 8080 microprocessor were written to conserve space and time. A proposed system for gamma ray spectrum analysis is described in hardware and software terms. The hardware for the proposed system would cost less than $9000 while commercial multichannel analyzers with the same capabilities would cost in excess of $23,000.

END

DATE
FILMED

11-81

DTIC